Sistema de paracaigudes per a ginys voladors

Per : Marc Rivero Prat

Escola Politècnica Superior d'Enginyeria de Manresa

EPSEM

Grau en Enginyeria de sistemes TIC

Treball presentat a: Pere Palà Schönwälder

Treball Final de Grau

Curs Acadèmic 2014-2015

Annex 1.

Escola Politècnica Superior d'Enginyeria de Manresa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

NORMATIVA DEL TFG DE L'EPSEM
Emplenar per l'estudiant/a

ANNEX 1.- AUTORITZACIÓ DE LA MATRÍCULA DEL TFG EN MODALITAT A o B

ESTUDIANT/A: MARC RIVERO PRAT **Núm. Identificatiu**_____
GRAU EN: ENGINYERIA DE SISTEMES TIC
DIRECTOR/A DEL TFG: PERE PALÀ
DEPARTAMENT: DIPSE
REALITZACIÓ EN ANGLÈS: Sí ☐ No ■
MODALITAT: A ☐ B ■

TÍTOL: SISTEMA DE PARACAIGUDES PER A GINYS VOLADORS

DESCRIPCIÓ:

Es desenvoluparà un sistema per desplegar un paracaigudes per protegir la integritat de diversos tipus de ginys voladors. A partir de la informació recollida de diversos sensors, es decideix el moment en què cal desplegar el paracaigudes per minimitzar els riscos d'una caiguda incontrolada.

SOL·LICITUD PER A SER AVALUAT DE LES COMPETÈNCIES GENÈRIQUES EN NIVELL 3 (si no estan assolides)	Sí	No
1.- Emprenedoria i innovació		X
2.- Sostenibilitat i compromís social		X
3.- Tercera Llengua		X
4.- Comunicació eficaç oral i escrita		X
5.- Ús solvent dels recursos d'informació		X
6.- Aprenentatge autònom		X
7.- Treball en equip		X

Director/a	Co-director/a (si s'escau)	Estudiant/a
Pere Palà Schönwälder		Marc Rivero
Signatura	Signatura	Signatura

Data Registre : _____ (validesa: un any des de la data del registre)

3 Exemplars: Director/a / Secretaria de l'Escola / Estudiant/a per incorporar al TFG

* Normativa Transitòria fins que l'aplicatiu PRISMA permeti la gestió telemàtica del TFG
(Document aprovat per la Comissió Permanent de 5 de juliol de 2012, i per la Junta de Centre de 12 de juliol de 2012)

Annex 2.

Escola Politècnica Superior d'Enginyeria de Manresa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

NORMATIVA DEL TFG DE L'EPSEM
Emplenar per l'estudiant/a

ANNEX 2.- LLIURAMENT DEL TFG

ESTUDIANT/A: MARC RIVERO PRAT **Núm. Identificatiu**
GRAU EN: ENGINYERIA DE SISTEMES TIC
DIRECTOR/A DEL TFG: PERE PALÀ SCHÖNWÄLDER
BREU RESUM DEL TFG (màxim 500 paraules):

El projecte tracta sobre la construcció i desenvolupament d'un paracaigudes per tal de salvaguardar un giny volador.

Aquest objectiu es vol aconseguir mitjançant un sensor baromètric que és el que recull les dades del món físic.

Inicialment s'estudien les lleis físiques que actuen sobre el conjunt del paracaigudes i les forces que si apliquen.

Continua explicant quins són els elements electrònics que formen part del mecanisme d'obertura, tals com: el sensor baromètric, l'Arduino o la memòria EEPROM entre d'altres.

Una de les parts més importants per a fer el projecte possible, és el software. Ja que aquest permet que les diferents parts del projecte treballin juntes. S'explica com el software treballa per tal d'aconseguir l'objectiu principal, salvaguardar el giny volador.

També s'expliquen alguns problemes ocorreguts durant el muntatge dels diferents components electrònics, com una placa de circuit imprès pot ajudar en això i també com crear la nostra pròpia placa.

Per a finalitzar s'explica el muntatge de tot el projecte en qüestió, la realització d'un experiment i les conclusions obtingudes de tot el treball.

PARAULES CLAU (entre 2 i 5):

Giny Volador Paracaigudes Sensor Baromètric

AUTORITZO A PUBLICAR EL TREBALL A UPCommons: Sí No

Director/a Pere Palà Schönwälder — Co-director/a (si s'escau) — Estudiant/a Marc Rivero

Signatura — Signatura — Signatura

Data lliurament : 8/7/2015

Lliurar a la secretaria: 1 còpia en paper de la memòria del TFG.
4 còpies en versió digital (CD o DVD) de la memòria del TFG

2 Exemplars: Secretaria de l'Escola / Estudiant/a

* Normativa Transitòria fins que l'aplicatiu PRISMA permeti la gestió telemàtica del TFG
(Document aprovat per la Comissió Permanent de 5 de juliol de 2012, i per la Junta de Centre de 12 de juliol de 2012)

Abstract

The objective of this project, is to develop a system for deploying a parachute in order to save the integrity of various flying widgets. This project was developed having in mind the idea of saving a drone. This goal wants to be achieved using the information given by a barometrical sensor.

The project itself starts explaining the physics behind the movement of the whole parachute. It talks about the different forces acting on the body, like the speed and acceleration achieved during the free fall, before and after deploying the parachute.

Then the shape of the parachute is also studied in order to obtaining the biggest drag force possible for going against the gravity force. The time of opening is also important so it will be explained.

the next part explains the device that makes the deployment of the parachute be possible, explaining all of the parts of this device. This parts are: the barometric sensor, the EEPROM memory and the Arduino Nano among others.

One of the most important parts for making this possible, is the software. The software allows to the different parts of the project work together. It is explained how the software works for achieving our primary goal: save the flying device attached to it.

This project, also explains some issues that occurred during the assembly of the different electronic components and how a printed circuit board can help on it. It also explain how to build one.

One of the last parts explained in the project is the assembly of the whole project, starting from the assembly of the mechanical parts up to the assembly of the electronic parts.

For ending the project, an experiment proving the efficiency of the parachute is done dropping the parachute from a high position in order to see if the parachute is deployed or not.

The last part of the project are the conclusions achieved during the realisation of the project, and the thanks to the people that has contributed on the project.

Index

1. INTRODUCCIÓ

Aquest projecte, ha vingut donat gracies a la construcció anterior d'un drone. El que ma portat a la realització d'aquet projecte es el fet de que tenia en ment fer quelcom relacionat amb el drone, però que no es tractes del drone en si mateix, ja que el treball de final de grau s'ha de fer individualment, i la construcció del drone la he realitzat conjuntament amb un company de la universitat.

A les hores, d'una llista d'idees per a portar a terme relacionades amb el drone, una relacionada amb la seguretat de l'aparell, em va cridar especialment la atenció i vaig decidir posar èmfasi sobre aquesta. Després de comentar-ho amb el meu tutor del projecte, vam decidir portar a terme aquesta idea, realitzar un paracaigudes per al drone.

No obstant aquest paracaigudes també pot servir per a altres ginys voladors, que siguin capaços de enlairar-se per si mateixos i arribin a una certa alçada, ja que per a que el desplegament del paracaigudes es faci efectiu, es necessita de certa altura.

La idea que s'ha seguit per al desenvolupament del projecte es senzilla i es la següent.

Primerament pensar en quin tipus de paracaigudes es vol fer, és a dir la forma i la grandària que tindrà aquest.

Seguidament pensar com es desplegarà el paracaigudes, i quin serà el mecanisme d'obertura automàtic que en permetrà l'alliberament.

També però no menys important vigilar quin serà el pes que podrà suportar el paracaigudes, procurant que la velocitat que prengui tot el conjunt sigui la menor possible abans d'impactar contra el terra amb l'objectiu d'evitar una caiguda aparatosa.

Per a finalitzar la introducció, cal resumir que aquest document recull l'explicació i desenvolupament de la construcció d'un paracaigudes pensat especialment per a un drone, ja que disposen d'una superfície suficientment ample, on es podria acoblar perfectament el prototip del paracaigudes, però que es aplicable a qualsevol tipus de giny volador.

2. INFORMACIÓ PRÈVIA NECESSÀRIA PER A LA COMPRESIÓ DEL PROJECTE

Aquí es farà un petit resum de possibles paraules per posar en situació al lector

Drone: Quan parlo de drone em refereixo a un quadcopter. Es a dir, un dispositiu volador que consta de una base amb quatre extremitats, i a cada una d'aquestes un motor amb la corresponent hèlix. Aquest dispositiu sol ser controlat mitjançant un comandament de RC.

I_2C: Circuit Inter integrat, es un bus serial síncron. Utilitza dos cables: SDA i SCL per a interaccionar amb el microcontrolador. La llibreria d'Arduino que utilitza s'anomena Wire.

Protoboard: Es una placa electrònica on si posen els diversos elements del muntatge a connectar. Com el seu nom indica, s'utilitza sovintment per a fer el muntatge del prototipus del circuit.

Beerware: Es un terme de llicencia de software atorgat sota termes molt lliures. Proveeix a l'usuari del codi, la llibertat per a fer el que li convingui amb el codi font d'aquest. Si l'usuari troba el codi del software útil, se li exhorta comprar a l'autor una cervesa per a tornar-li el favor. Aquesta paraula va ser inventada per John Bristor a Penascola, Florida el 25 d'abril de 1987.

Sentinella: En programació, un valor de sentinella (també anomenat valor de flag), es un valor especial en el context d'un algoritme, el qual es fa servir com a condició de finalització típicament en un algoritme que es vagui repetint cíclicament.

Baud rate: Es numero d'unitats de senyal per segon. Un baud pot contenir varis bits, per tant no s'ha de confondre amb el bit rate, que es la tassa de bits per segon.

3. ESTUDI DEL MON FÍSIC (CAIGUDA LLIURE)

Per tal de conèixer el comportament del conjunt del paracaigudes, hem d'estudiar el mon físic que ens envolta i això comporta saber sobre les equacions del moviment relacionades amb la caiguda de cossos.

Estudiar la caiguda lliure del conjunt del paracaigudes es important ja que així es pot saber la velocitat a la qual el paracaigudes es desplegarà i els metres que recorrerà abans que això succeeixi.

La idea per al desplegament, esta pensada de tal forma que es produirà instants després que el conjunt del paracaigudes detecti que esta en caiguda lliure.

Es suposa que quan els motors del Drone estan en un funcionament correcte, l'aparell no entra en cap moment en caiguda lliure, ja que descendeix a una velocitat constant i per tant el paracaigudes no entra en acció.

La idea per al desplegament succeeix de la següent manera:

1. Suposem que l'aparell a arribat al punt més alt del seu vol, i això es pot donar per varis motius:
    - Que la bateria agües esgotat la seva carrega.
    - Que un dels motors falles.
    - O simplement si es tractes d'un coet l'acceleració i velocitat per un instant es convertissin en 0 després d'arribar al punt més alt del seu vol.

    Es a les hores, quan es denomina que hem entrat en caiguda lliure.

2. Existeix un petit espai de temps entre que comença la caiguda lliure i es té coneixement d'aquesta per part del conjunt de sensor baromètric i l'Arduino. El sensor baromètric a la que detecta una diferencia de pressió alta en un breu espai de temps, decideix enviar un senyal elèctric al mecanisme d'obertura.

3. Un cop obert el sistema del paracaigudes, cal fer el desplegament adequat del propi paracaigudes i això requereix de cert temps i alçada. Al ser un desplegament mecànic degut a la molla, no es instantani i a pocs metres del terra es podria donar el cas de que no li donges temps per al desplegament.

4. Finalment si el desplegament ha anat com s'espera, la velocitat de caiguda es veura reduida fins a arribar a una velocitat limit on apartir d'aqui sera constant fins a tocar terra.

Anem a veure les forces que actuen durant tot el procés, i com es poden mesurar.

Suposem que el conjunt ja ha arribat a l'altura màxima tal com s'ha explicat abans i comença la caiguda lliure. L'única força que actua sobre el conjunt es la de la gravetat, menyspreant el

fregament amb l'aire ja que aquest es petit, l'acceleració es constant i la velocitat en el instant en que comença a caure es de 0 però ràpidament varia en funció del temps.

Les formules de la velocitat i de l'acceleració són les següents:

$$V = -g \cdot T$$

$$a = -g$$

L'acceleració es la força de la gravetat directament. Es negativa perquè estem treballant sobre l'eix de les X mirant cap amunt, es a dir portarà una acceleració de -9,81m/s^2. Mentre que la velocitat ve donada en funció del temps ja que com més temps passi major serà la velocitat que porti. També podem determinar la posició en funció del temps, si coneixem la posició des de la que ha començat la caiguda lliure. La formula que s'usa es la següent:

$$X = X_o - g \cdot \frac{T^2}{2}$$

Per el contrari, també podem determinar quin es el temps que a transcorregut per arribar des d'una posició a l'altre, o simplement el temps que trigarà en recorre uns metres determinats.

Un cop s'ha obert el paracaigudes, les equacions del moviment varien. El conjunt esta sotmès a l'acció del seu pes, i la força de fregament es proporcional al quadrat de la velocitat.

$$m \cdot a = m \cdot -g + k \cdot v^2$$

La constant de proporcionalitat k es la següent:

$$k = \frac{r \cdot A \cdot d}{2}$$

- **r** es correspon a la densitat del aire. Encara que la densitat del aire varií en funció de la altura, jo tindre en compte el seu valor a nivell del mar ja que l'aparell no s'enlairarà tant com per variar. Aquest valor es 1,29 kg/m^3.
- **A** es l'àrea de la secció transversal frontal exposada al aire, en altres paraules i en el cas que ens ocupa, l'àrea en metres quadrats del paracaigudes.
- **d** es un coeficient que depèn de la forma del objecte, i varia en funció de la resistència que ofereix al vent.

Forma del objecte	**Valor aproximat de *d***
Disc circular	1.2
Cub	1.05

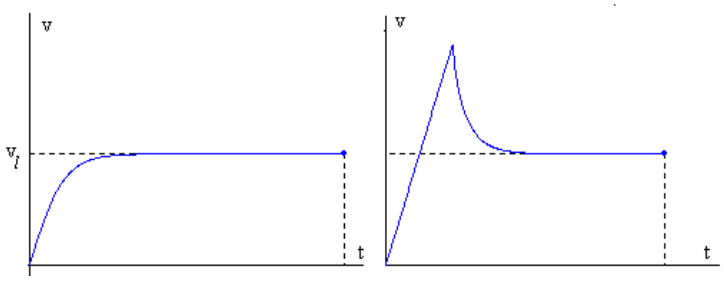
Esfera	0.4
Cos aerodinàmic	0.04

Com que el conjunt del paracaigudes es menys aerodinàmic que una esfera però més aerodinàmic que un disc prenc com a referència el promig dels valors donats per la taula anterior, es a dir **d** = 0.8

Quan es desplega el paracaigudes, es troba en caiguda lliure i es redueix bruscament la velocitat fins a arribar a una velocitat límit constant v_l, que s'obté quan el pes es igual a la força de fregament, es a dir quan l'acceleració es zero.

$$0 = m \cdot -g + k \cdot v^2$$

El valor d'aquesta velocitat límit, es independent de la velocitat inicial del nostre conjunt del paracaigudes en el moment d'obertura, tal i com s'observa a les següents figures.



En el primer gràfic podem apreciar que el paracaigudes s'obre abans de sobrepassar la velocitat límit i s'accelera el conjunt a menor ritme fins a arribar a la velocitat límit, que es mantindrà durant tota la caiguda. Per altre banda si el paracaigudes s'obre un cop superada la velocitat límit el conjunt anirà reduint la seva velocitat fins arribar a aquesta.

La formula de la velocitat límit es la següent:

$$v_l = \sqrt{\frac{m \cdot g}{k}}$$

L'equació del moviment quan s'ha obert el paracaigudes es pot escriure de la següent manera

$$\frac{dv}{dt} = -g + \frac{k}{m}v^2$$

Integrem l'equació del moviment per obtenir la velocitat v del mòbil a qualsevol instant t. Les condicions inicials son: v_0 que es la velocitat del conjunt en el instant t_0 en el que s'obre el paracaigudes.

$$\int_{v_0}^{v} \frac{dv}{-g + \frac{k}{m}v^2} = \int_{t_0}^{t} dt$$

Per a integrar es fa el canvi $v = z \cdot v_l$.

$$\frac{v_l}{g}\int_{z_0}^{z} \frac{dz}{z^2 - 1} = \frac{v_l}{2g}\left(\int_{z_0}^{z} \frac{dz}{z-1} - \int_{z_0}^{z} \frac{dz}{z+1} - = \frac{v_l}{2g} ln \frac{(z-1)(z_0+1)}{(z+1)(z_0-1)}\right)$$

Es desfà el canvi i s'aclareix v en funció del temps ($t-t_0$). Després de fer unes operacions la expressió resultant es la següent:

$$v = -v_l \frac{(v_0 - v_l)\exp\left(\frac{g}{v_l}(t - t_0)\right) + (v_0 + v_l)\exp\left(\frac{-g}{v_l}(t - t_0)\right)}{(v_0 - v_l)\exp\left(\frac{g}{v_l}(t - t_0)\right) - (v_0 + v_l)\exp\left(\frac{-g}{v_l}(t - t_0)\right)}$$

Podem obtenir també la expressió de la posició del paracaigudes en funció de la velocitat, fent un canvi de variable

$$\frac{dv}{dt} = \frac{dv}{dx}\frac{dx}{dt} = \frac{dv}{dx}v$$

Aleshores la equació del moviment es transforma en:

$$v\frac{dv}{dx} = -g + \frac{k}{m}v^2$$

Que es pot integrar de forma immediata.

$$\int_{v_0}^{v} \frac{vdv}{-g + \frac{k}{m}v^2} = \int_{x_0}^{x} dx$$

L'altura **x** del nostre conjunt en funció de la seva velocitat **v** es:

$$x - x_0 = \frac{v_l^2}{2g} ln \frac{v^2 - v_l^2}{v_0^2 - v_l^2}$$

Si aïllem la velocitat v en funció de la posició x del conjunt del paracaigudes, obtenim la següent formula:

$$v^2 = v_l^2 + (v_0^2 - v_l^2)\exp\left(-\frac{2g}{v_l^2}(x_0 - x)\right)$$

Anem a veure un exemple pràctic de les formules vistes fins ara.

3.1. EXEMPLE

Anem a suposar un exemple on es facin servir les formules anteriors, per saber a quina velocitat baixaria el conjunt del paracaigudes.

- El conjunt del paracaigudes sumat al pes del drone té un pes aproximat de m=1,5kg, el drone pesa 0.85 kg mentre que el conjunt del paracaigudes pesa 0,65 kg.
- L'àrea del paracaigudes, serà la del prototip del paracaigudes 2,617m^2
- Suposarem que el conjunt del paracaigudes comença el seu descens als 200 m
- I el paracaigudes s'obre als 180 metres.

Calcularem la velocitat a la que arribarà al terra tenint en compte que

- La densitat del aire es de 1,29 kg/m^3 .Que es la densitat del aire a nivell de mar i a temperatura ambient.
- Mentre que el coeficient de forma com ja em explicat abans serà de 0,8.

Calculem v_l

$$k = \frac{1{,}29 \cdot 2{,}617 \cdot 0{,}8}{2} = 1{,}35 \qquad v_l = \sqrt{\frac{1{,}5 \cdot 9{,}81}{1{,}35}} = \boldsymbol{3{,}3 m/s}$$

Tenim dons que la velocitat límit del nostre conjunt serà de 3,3 m/s

Ara anem a veure la velocitat a la qual es desplegarà el paracaigudes, suposant que es desplega als 180 metres.

$$180 = 200 - 9{,}8 \cdot \frac{t^2}{2} \qquad t = \sqrt{\frac{-40}{-9{,}8}} = 2{,}02s$$

$$v = 9{,}8 \cdot t = \boldsymbol{19{,}79 m/s}$$

la velocitat a la que arriba als 180 metres es de 19,79 m/s que es el moment en el que es desplega el paracaigudes. Anem a veure quina es la velocitat final.

$$v^2 = 3{,}3^2 + (19{,}79^2 - 3{,}3^2)\exp\left(-\frac{2 \cdot 9{,}8}{3{,}3^2}(198 - 0)\right)$$

$$v^2 = 10{,}89 + 380{,}75\exp(-356{,}36)$$

$$v = \sqrt{10{,}89} = \boldsymbol{3{,}3 m/s}$$

La velocitat a la qual el conjunt tocarà terra serà de 3,3 m/s, es una velocitat suficientment segura per a que no es faci mal be cap dels components del drone o l'aparell en qüestió.

La velocitat final sempre serà la mateixa, i es la velocitat límit. No obstant la velocitat que si canvia, es la velocitat a la qual s'obrirà el paracaigudes, i aquesta com em vist be donada pel temps que passi abans de l'obertura. Si aquesta velocitat fos molt gran es podria donar que els materials del paracaigudes no aguantessin la estrebada i es trenques, però com esta pensat per a que s'obri al moment que detecti la caiguda lliure, no passarà massa temps i per tant la velocitat serà petita.

4. TELA DEL PARACAIGUDES

La tela del paracaigudes es gairebé l'element més important de tot el conjunt del paracaigudes, ja que aquesta evitarà que descendeixi a gran velocitat i tingui un descens controlat. Permetent així recuperar el objecte que es vol salvaguardar.

La tela del paracaigudes normalment esta feta d'un material lleuger i flexible per assegurar l'èxit de la caiguda sense que el material s'estripi o tingui un mal funcionament a l'hora de fer el descens. Aquesta, a de ser una tela que s'obri a mesura que el conjunt va caient, augmentant lleugerament la seva area durant la caiguda.

Com més gran sigui l'àrea que esta amb contacte amb l'aire, més arrossegament tindrà i això es tradueix a una major resistència a l'hora de baixar fent que el conjunt es desplaci més a poc a poc.

L'aire que entra en contacte amb la tela, crearà una força que actuarà en direcció contraria a la de la gravetat, causant que la velocitat descendeixi fins a arribar a la velocitat límit que es quan ja no hi ha acceleració, permetent al conjunt baixar a una velocitat constant fins a tocar terra.

4.1. EXPERIMENT

He trobat per Internet un petit experiment que determina l'efecte que té la forma del paracaigudes sobre el temps que tardarà en descendir.

Per a fer això han creat varis paracaigudes amb una mida constant. En altres paraules que l'àrea del paracaigudes es la mateixa, però amb diferents formes.

Les formes que han utilitzat són les següents:

- Cercle
- Triangle
- Quadrat
- Pentàgon
- Hexàgon

On cada un d'aquestes formes té una area de 0,04m^2.

A la hipòtesi de l'experiment, es diu que el cercle serà el que descendirà més a poc a poc ja que té més força d'arrossegament i oferirà més resistència a l'aire.

Per a fer l'experiment han tallat 24 fils d'uns 20 cm de llargada i els han lligat un a cada un dels vèrtex de les diferents formes, a excepció del cercle que tindrà 6 fils espaiats equitativament. al final dels fils hi han lligat un clip per a que faci de pes.

L'experiment l'han dut a terme en una habitació sense corrents d'aire per evitar que això afectes al rendiment de la baixada. Aquest experiment s'ha repetit set vegades per a cada un dels diferents models de paracaigudes, i els resultats han estat els següents:

Nombre de costats	Prova 1	Prova 2	Prova 3	Prova 4	Prova 5	Prova 6	Prova 7	Mitjana
3 (triangle)	3,5	3,1	3,4	3,2	3,1	2,9	3,3	3,21
4 (quadrat)	4,4	3,8	3,1	3,3	4,2	3,4	3,9	3,73
5 (pentàgon)	3,9	4,0	3,9	4,1	3,8	4,2	3,9	3,97
6 (hexàgon)	4,6	4,3	3,9	3,5	4,6	4,2	3,6	4,10
0 (cercle)	4,2	4,2	4,5	4,1	4,0	3,8	4,3	4,16

Els resultats estan en segons.

Com podem observar als resultats de la prova, a mesura que augmenta el nombre de costats el temps de caiguda també va en augment, amb l'excepció del cercle que no té cap costat. Tot i aixi és el que té la caiguda més llarga.

També es de sentit comú pensar que una caiguda més lenta es correspon a un paracaigudes més eficaç.

Finalment, la conclusió que n'extreuen dels diferents resultats es que a mesura que el paracaigudes va adquirint una forma més circular, al afegir costats, el temps de caiguda també augmenta. També acabar dient que podrien haver afegit més formes al experiment per a millorar-lo, així com un octàgon.

4.2. ELECCIÓ DE LA FORMA

Tornant al nostre paracaigudes i veient els resultats de l'experiment anterior, crec que la millor opció per a la forma del paracaigudes hauria de ser entre l'hexàgon i el cercle. Per això l'opció escollida a estat una forma octagonal.

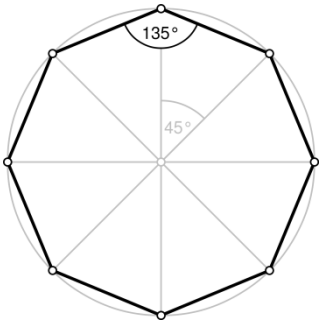
Aquesta forma es ideal per al disseny d'un paracaigudes. Com ja em dit per que té un descens lent i suau apart que es pot trobar amb suma facilitat. Podem trobar una tela octogonal en un objecte tan comú com es el paraigües. Aquests, estan composats d'una tela lleugera, raonablement flexible, resistent a l'aigua i també al vent.

Així doncs per al meu prototip del conjunt del paracaigudes, he utilitzat un vell paraigües de pesca, que es ideal per a frenar la caiguda de tot el conjunt.

El paracaigudes en qüestió esta fet 100% de poliester, ja que es un material econòmic, té molta resistència i resiliència, pesa poc, hidrofòbic i té un punt de fusió molt elevat.

4.3. GRANDÀRIA

Per tal de trobar l'àrea total del paracaigudes que té forma octagonal, podem dividir l'octàgon en vuit triangles isòsceles.



El següent pas es fixar-se en quina és la longitud des de qualsevol dels vèrtexs al centre d'aquest i en la longitud que fa cada un dels costats.

Els costats del octagon fan 68,5cm, mentre que del vèrtex al centre fa 101,5cm. Ara si apliquem el teorema de Pitàgores podrem saber quina es la distancia que hi ha del centre de l'octàgon al centre de qualsevol dels costats, ja que la meitat del triangle isòsceles es un triangle rectangle.

$$101{,}5 = \sqrt{(\frac{68{,}5}{2})^2 + x^2}$$

$$x = 95{,}54\ cm$$

Sabent que cada un dels triangles isòsceles forma un rectangle, passem a calcular l'àrea d'aquest.

$$Area\ Triangle = \frac{68{,}5}{2} \cdot 95{,}54 = 3.272{,}24\ cm^2$$

I un cop tenim l'àrea d'un dels triangles podem obtenir l'àrea total del octàgon en centímetres quadrats.

$$Area\ Octagon = Area\ Triangle \cdot 8 = 26.177{,}9\ cm^2$$

Passat a metres:

$$\frac{26.177{,}9}{10.000} = 2{,}617 m^2$$

Així sabem que l'àrea total de la tela del paracaigudes es de 2,617m^2. Es una tela bastant gran, però d'aquesta manera aconseguim reduir la velocitat de caiguda notablement, i així poder assegurar la integritat del dispositiu després de la caiguda. La velocitat de caiguda amb paracaigudes es una velocitat constant mentre que sense es va accelerant fins a tocar el terra, com s'explica al apartat de caiguda lliure.

No obstant la grandària del paracaigudes quan esta ben enrotllat, s'assembla a un petit cilindre que té unes dimensions de 13cm de llargada i 8,5cm de diàmetre.

4.4. TEMPS D'OBERTURA

Per a que la tela sigui efectiva com a paracaigudes, requereix que tota la superfície del paracaigudes estigui completament oberta i ben estirada. Si pel contrari no fos així, la velocitat de caiguda del conjunt es veuria afectada negativament. En altres paraules, la velocitat del conjunt seria major, amb les conseqüències d'una caiguda aparatosa i fatal per al dispositiu a salvaguardar.

El mecanisme que treu la tela del paracaigudes fora del seu recipient, es una molla compresa que té un temps de reacció per a retornar a al seva posició inicial (posició no compresa). A aquest temps se li ha de sumar el temps del desplegament de la tela, i depenent de les condicions meteorològiques, tal com el vent pot trigar des de 1 fins a 3 segons.

Si el conjunt no es troba a una altura suficientment elevada aquests segons poden significar que s'estavelli, ja que tot i que es parteix d'una altura determinada on la velocitat es 0, el sensor s'ha d'adonar de que esta caient. A les hores el mecanisme ha d'obrir la tapa per a que la molla empenyi el paracaigudes cap enfora i finalment s'obri la tela.

La longitud de les vuit tires que estan lligades a cada un dels vèrtex de la tela del paracaigudes tenen una longitud lleugerament superior al diàmetre de la tela permetent així que el paracaigudes s'obri en la seva màxima extensió. Si mirem el paracaigudes de perfil, i nomes agafem dos de les 8 cordes en vèrtex oposats, la figura que dibuixa es similar a la d'un triangle isòsceles.

Cal dir que les vuit cordes del paracaigudes, estan lligades a un anell d'alumini situat al final de les cordes del paracaigudes. I a la vegada aquest té 3 cordes més que van lligades a la base del recipient del conjunt del paracaigudes, on cada corda es capaç de suportar un pes total de 14Kg.

5. DISPOSITIU D'OBERTURA AUTOMÀTICA

Per tal que el paracaigudes s'obri en el moment necessari, es necessita d'un aparell que sigui capaç de determinar per ell mateix el moment d'obertura idoni.

Per aquest motiu he creat una capseta que conte els diferents elements per a realitzar aquesta tasca. Els elements són els següents:

- **Microcontrolador:** Es la peça central i unitat de control del sistema. Es tracta de l'Arduino Nano que es el que gestionarà les dades rebudes pel sensor, i s'ocuparà d'enviar el senyal al servo motor per tal d'alliberar el paracaigudes.

- **Servomotor:** Es l'element encarregat d'alliberar la tapa on esta plegada la tela del paracaigudes. Quan aquest rep el senyal enviat pel microcontrolador, es posiciona en la posició pertanyent alliberant-la.

- **Baròmetre:** S'ocupa de recollir les dades de la pressió atmosfèrica i enviar-les al microcontrolador.

- **Bateria:** Proporciona l'energia elèctrica a tots els components electrònics.

- **Interruptor:** Posa en funcionament el circuit, o per el contrari el desactiva.

- **EEPROM:** Es la memòria externa on es guarden les dades dels diferents canvis de pressió.

Aquesta capseta va adossada al recipient del paracaigudes. La capseta té les següents dimensions: 87x64x28mm.

Si el prototipus s'arribés a comercialitzar, el contingut de la capseta aniria dintre del propi recipient del paracaigudes, però per a fer proves i poder canviar el software constantment de l'Arduino em resultava més pràctic deixar-ho així.

5.1. ALIMENTACIÓ

L'alimentació és una part important del conjunt del paracaigudes i també s'ha de tenir en compte.

La placa que alimentarem es una placa de desenvolupament, anomenada Arduino Nano. El voltatge recomanat per la qual esta entre 7 i 12 volts.

Com que el voltatge és el mencionat anteriorment, utilitzo una pila de 9v, que em proporciona l'energia suficient per alimentar tot el conjunt dels elements electrònics.

La pila esta connectada a un interruptor que és el que dona pas al corrent elèctric de tot el circuit, aquest és l'interruptor general per que el dispositiu d'obertura automàtica s'accioni o

no. Apart també actua com a reset del sistema, ja que al deixar l'Arduino sense corrent, quan el torna a rebre comença a executar el programa de nou.

Un cop l'Arduino esta sent alimentat, aquest també ha de proporcionar voltatge a tres dispositius més què són els següents:

- Sensor baromètric: Alimentat a 3,3v per una de les sortides de l'Arduino i el propi GND d'aquest.

- Servomotor: Esta connectat a 5v i al GND en la placa de l'Arduino.

- EEPROM: De la mateixa manera que el servomotor, esta connectat a 5v i al GND.

5.2. SENSOR BAROMÈTRIC

5.2.1. Que es un sensor baromètric?

Un sensor baromètric, al igual que un baròmetre, és un component electrònic capaç de detectar les pressions atmosfèriques. La pressió atmosfèrica és el pes per unitat de superfície exercida per l'atmosfera.

5.2.2. Perquè s'ha utilitzat un sensor baromètric?

Per a la realització d'aquest projecte es necessitava d'un o varis sensors per tal de detectar quin era el moment idoni en el qual s'avia de desplegar el paracaigudes. A les hores buscant per Internet he pres la decisió d'utilitzar aquest tipus de sensor, ja que es molt utilitzat en l'aviació. Per saber si es puja o es baixa, es realitzen varies mesures de la pressió atmosfèrica.

El sensor baromètric es molt utilitzat en l'aviació i tots els models actuals d'avions comercials en porten al menys un per a detectar la altura o la pressió atmosfèrica.

Al any 1940 Leonid Vladimirovich Savichev, enginyer rus, va crear un sistema de paracaigudes automàtic que permetia obrir el paracaigudes un cop el paracaigudista es trobava a una alçada determinada, de exactament 4267m. Si el pilot de l'avio s'equivocava d'alçada al llençar els paracaigudistes o els propis paracaigudistes podien quedar inconscients per a qualsevol cosa, el dispositiu s'obriria automàticament salvant així la vida del paracaigudista.

Aquest enginyer rus, es va veure motivat a crear aquest dispositiu quan el seu govern va llençar a concurs la creació d'un sistema d'obertura automàtic per als paracaigudistes. Ja que feia relativament poc una jove paracaigudista va morir després de que el seu paracaigudes no se li obris correctament.

El sensor que va utilitzar era un sensor baromètric amb càmera aneroide, aleshores vaig començar a buscar informació sobre els sensors baromètrics, i vaig veure que era una elecció molt encertada per a detectar el moment exacte en el que s'ha d'obrir el paracaigudes.

El sensor utilitzat però no es de càmera aneroide, sinó que es un sensor de pressió atmosfèrica digital de alta precisió.

5.2.3. Com es el sensor?

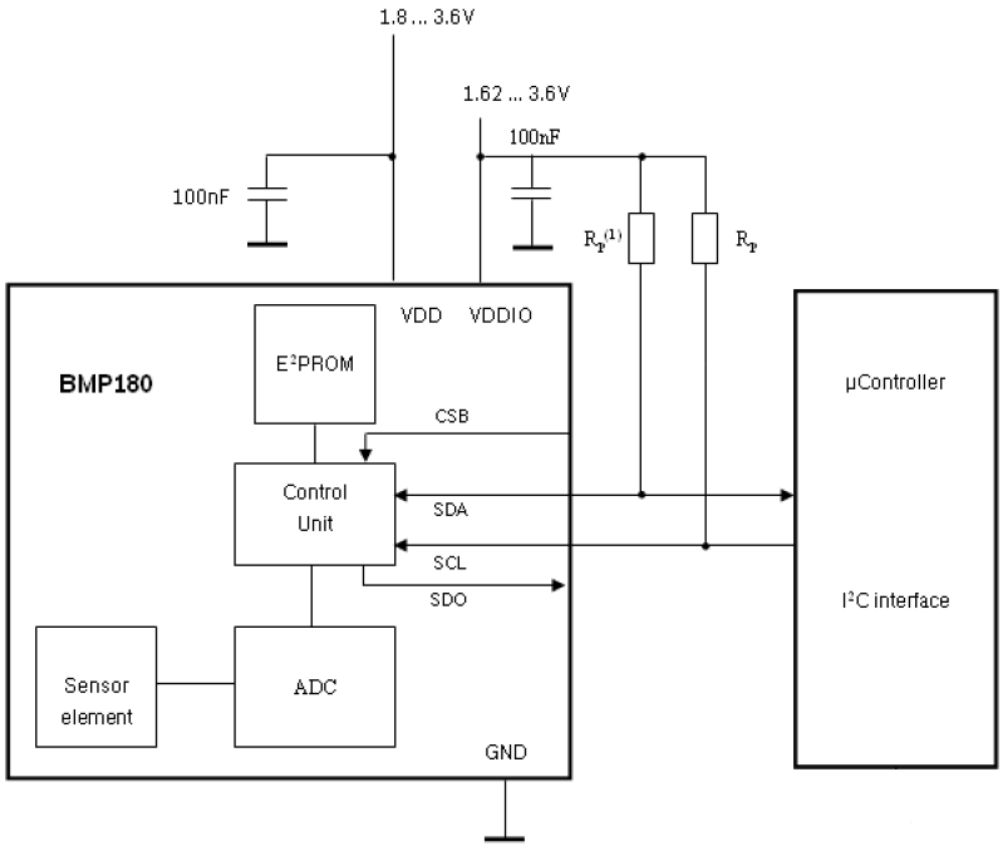
El sensor baromètric que he utilitzat per al projecte en qüestió, es un petit dispositiu baromètric fabricat per la casa Bosch, i muntat sobre una plaqueta de circuit imprès fabricada per Sparkfun.

Te un rang de mesura en Questions d'altitud que van des de +9000m fins a -500m(en relació amb el nivell de mar) que equivaldrien a 300hPa fins a 1100hPa. A major altura menor es la pressió atmosfèrica ja que hi ha menys volum d'aire per sobre i això es reflexa en una menor pressió.

El model que ens ocupa és el BMP180 que pot treballar entre temperatures de -40ºC a 85ºC i pot operar entre un rang de voltatges des de 1,8V fins als 3,6V. Com que opera entre aquests voltatges, he utilitzat la sortida de 3,3V que ofereix l'Arduino per al voltatge positiu. Si el voltatge proporcionat al sensor baromètric fos superior als 3,6V es podria fer malbé el dispositiu, i podria deixar de funcionar.

El sensor, consisteix d'un sensor piezo-resistiu, un convertidor d'analògic a digital i d'una unitat de control amb una EEPROM i interfície serial I_2C.

El sensor piezo-resistiu es el que obté les dades del mon físic en format analògic, a les hores passa aquestes dades pel convertidor d'analògic a digital, fins a la unitat de control, emmagatzema fins a 176 bits d'aquestes dades, que posteriorment seran utilitzades per a compensar el offset, la temperatura o altres paràmetres. Finalment les dades seran enviades pel bus I_2C.



Com podem apreciar en la imatge el sensor baromètric té varies sortides i entrades físiques que són les següents:

- VDD: Es la entrada de voltatge positiu i es la que jo faig servir, proporcionant-li un corrent de 3,3v mitjançant la sortida corresponent del micró controlador Arduino.

- VDDIO: Es una entrada alternativa del baròmetre, que jo no utilitzo, que esta connectada mitjançant dues resistències de 4,7kΩ a la interfície I_2C , i que serveix per alimentar igual el baròmetre. La principal diferencia recau en que permet alimentar el baròmetre amb una tensió més baixa d'entrada.

- GND: Es el retorn o els 0v, s'utilitza per a que el circuit estigui tencat i així pugi circular el corrent.

- SDA: Un dels canals del bus I2C per on passen les dades, per anar i tornar. En el meu cas el connecto al pin A4 de l'Arduino.

- SCL: Es el clock del port I2C, es connecta al pin A5 de l'Arduino.

En altres paraules, nomes utilitzo quatre dels cinc ports per a rebre i enviar dades i alimentar el sensor que son: VDD, GND, SDA i SCL.

5.2.4. Com funciona?

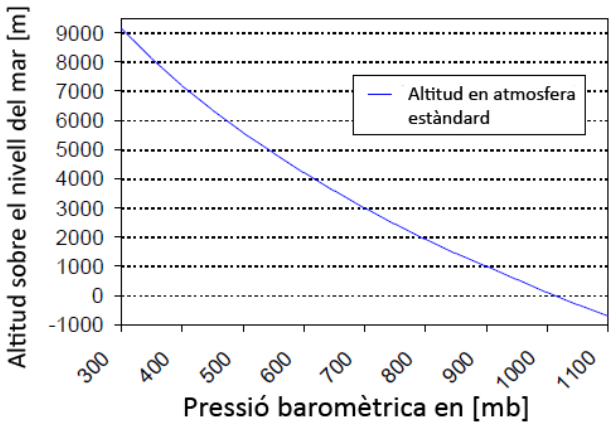
Tots els models de sensors baromètrics es veuen afectats pel medi que els envolta, pressió atmosfèrica, pluja, altura.

El sensor baromètric que ens ocupa, el BMP180, es veu afectat per coses tan normals com pot ser la llum directe del sol. Es a dir, si el dispositiu piezoelèctric que es l'encarregat de transformar els valors del mon físic en dades reals, es troba en una zona on no li dona la llum del sol directament, i de cop i volta entra en una zona en exposició directe amb la llum solar, els valors que donarà no seran els correctes i aquest petit fet podria fer que el paracaigudes s'accionés quan no li toca.

Per tant a la resolució que he arribat jo per tal de solucionar el problema anterior, es la de introduir el sensor baromètric dins d'una caixeta opaca per tal de que no es vegi afectat per la llum solar. Tot i així al estar situada dintre de una caixa es pot pensar que els valors que obtindrà el sensor baromètric no seran els mateixos que els que podria obtenir si es trobes a fora de la capseta, però això es total ment fals. Ja que el conjunt de la caixa es veu pressionat pel mateix volum d'aire que si estigues fora de la capseta, i els valors varien molt poc, permetent al sensor detectar la mateixa pressió atmosfèrica. Això és degut a que la capseta no es totalment estanca.

No obstant, el valor de la pressió atmosfèrica no ens interessà pròpiament dit. Es a dir si que necessitem un valor de referència, però que el valor sigui de 950 ó 798 no és de vital importància. Seria un valor important en el cas de que volguéssim mesurar l'altura en la qual es troba el nostre dispositiu.

Per exemple si ens fixem en el següent gràfic.



Observem que es pot determinar l'altura a la que ens trobem mitjançant la pressió atmosfèrica que obtenim com a dada.

També es podem determinar l'altura o la pròpia prensió atmosfèrica coneixent la pressió atmosfèrica a nivell de mar amb la següent formula baromètrica internacional.

$$Altitud = 44330 * \left(1 - \left(\frac{P}{P_o}\right)^{\frac{1}{5,255}}\right)$$

Un cop coneguda la pressió atmosfèrica a nivell de mar P_o, es pot substituir el valor de l'altitud per a conèixer la pressió atmosfèrica que hi haurà en aquella altitud. O a la inversa podem substituir la pressió que tinguem actualment i conèixer així a quina altura ens trobem sobre del nivell del mar.

El dia 10 d'abril a les 11 del mati, vaig consultar la pagina web de l'agencia estatal de meteorologia (AEMET), on es pot saber la pressió atmosfèrica a diferents punts de territori espanyol. Vaig agafar dos valors en concret per a la demostració de la formula i són els següents:

- Barcelona (Aeroport), es troba situat a una altura de 4m sobre el nivell del mar i la pressió atmosfèrica d'aquell dia era de 1025,7mb
- Manresa, es troba situada a una altura de 291m sobre del nivell del mar i la pressió atmosfèrica era de 991,8mb

Amb les dades de l'aeroport de Barcelona ja que es troba més a prop del nivell del mar, trobarem la pressió atmosfèrica a nivell d'aquest.

$$4 = 44330 \cdot \left(1 - \left(\frac{1025,7}{P_o}\right)^{\frac{1}{5,255}}\right)$$

$$P_o = 1026,18$$

Un cop trobada la pressió atmosfèrica a nivell del mar podem trobar l'altura de Manresa, amb la pressió atmosfèrica d'aquesta.

$$Altitud = 44330 * \left(1 - \left(\frac{991,8}{1026,18}\right)^{\frac{1}{5,255}}\right)$$

$$Altitud = 286m$$

Com es pot apreciar hi ha una petita variació de 5m però es bastant exacte, ja que les condicions climàtiques de Barcelona a Manresa són diferents i poden afectar a aquesta petita diferencia.

Aquest mètode s'utilitza en el projecte per a determinar l'altura a la qual s'ha obert el paracaigudes, i posteriorment desar-la a la memòria EEPROM. També tinc en compte la pressió atmosfèrica cada certs instants de temps per determinar si l'aparell es troba en caiguda lliure i sense control o no.

La idea funciona de la següent manera. Com que sabem que a major altura la pressió es menor vaig guardant els valors que va obtenint el sensor i faig una resta amb el valor pres anteriorment del actual

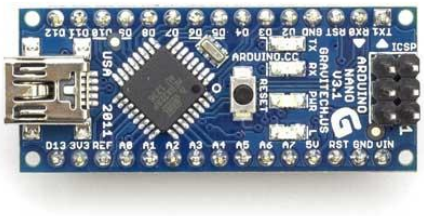
$$Resultat = Valor\ actual - Valor\ anterior$$

Si aquest resultat es positiu significa que esta descendint, però si el resultat es prou gran significarà que esta baixant a gran velocitat i per tant en caiguda lliure i sense control. Es en aquest moment quan el paracaigudes s'hauria d'obrir.

Per altra banda si el resultat d'aquesta resta es negatiu, significa que estem pujant i aquí no hi perill.

5.3. MICROCONTROLADOR

El microcontrolador que he utilitzat és el que porta l'Arduino Uno i també l'Arduino Nano. L'Arduino utilitzat, a estat el Nano per al prototip final, però al inici de les proves s'utilitzava el UNO. Ambdós utilitzen el mateix microprocessador, estem parlant del ATmega328. Aquest es el propi Microcontrolador en si mateix, però gracies a la placa Arduino es més fàcil de treballar sobre el processador.



Si ens fixem en el NANO té 14 entrades/sortides digitals a les quals si pot accedir fàcilment gracies als pins, 6 de les quals poden ser utilitzats com a sortides PWM (Pulse Wide Modulation). Treballa a una freqüència de 16MHz, utilitza un port USB ja sigui per a obtenir corrent o per a inserir el software amb el que treballarà, capçalera ICSP i un boto per tornar a inicialitzar(reset).

5.3.1. Alimentació

La font d'alimentació es selecciona automàticament, ja sigui per USB o per una font externa d'energia (una bateria per exemple).

L'energia proporcionada per una bateria pot ser inserida mitjançant els pins Vin i Gnd del propi Arduino.

La placa pot operar amb una font d'energia de 6 a 20 volts. No obstant si l'energia proporcionada a la placa es menys de 7V, l'energia que proporcioni la placa pel pin de 5V podria ser menor a cinc i podria ser inestable. Si pel contrari s'utilitzen més de 12V, el regulador de voltatge es podria sobre escalfar i fer malbé la placa. Així que el voltatge recomanat es de 7 a 12 volts.

Els pins del voltatge són els següents:

- Vin: Es el pin que s'utilitza per entrar voltatge a l'Arduino amb una font de alimentació externa. Aquesta alimentació ha de ser de 7 a 12V.
- 5V: Aquest pin ofereix una sortida regulada de 5V. Alimentar la placa per al pin de 5V o el de 3,3V podria fer malbé la placa ja que passa pel regulador de voltatge.
- 3,3V: De igual manera que el de 5V, proporcionà 3,3V. La intensitat màxima que dona es de 50mA.
- Gnd: són els pins de terra.

5.3.2. Memòria

El ATmega328 té 32KB de memòria flash, dels quals 0,5KB són utilitzats pel carregador d'arrencada (bootloader). També disposa de 2KB de SRAM i 1KB de EEPROM(la qual es pot llegir i escriure amb la llibreria EEPROM).

5.3.3. Input and Output

Cada un dels 14 pins de l'Arduino poden ser utilitzats com a entrada o sortida. Utilitzant les funcions següents: pinMode(), digitalWrite() i digitalRead(), actuant a 5 volts. Cada pin pot produir o rebre un màxim de 40 mA i disposen d'una resistència de pull-up, desconnectada per defecte, entre 20 i 50 kOhms.

A mes, alguns pins tenen funcions especialitzades:

- Serial: 0 (RX) i 1 (TX): Utilitzats per rebre i transmetre respectivament informació serial sobre el TTL (Time To Live).
- Interrupcions externes, 2 i 3: Aquests pins poden ser configurats per fer saltar una interrupció en un valor baix, un flanc ascendent o descendent, o un canvi de valor.
- PWM: 3,5,6,9,10 i 11: Proveeixen 8-bits de sortida PWM amb la funció *analogWrite()*
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK): Aquests pins suporten la comunicació SPI utilitzant la corresponent llibreria.
- LED:13: La pota 13 es correspon al led que esta integrat a la pròpia placa quan el pin té un valor alt el LED esta encès, quan el pin té un valor baix, el LED esta apagat.

L'Arduino té 8 entrades analògiques, marcades des de A0 fins A7, cada una de les quals proveeix 10 bits de resolució. Per defecte mesuren des de 0 fins a 5 volts, però es possible canviar això, utilitzant la funció analogReference().

Els pins A4 i A5, també anomenats SDA i SCL, suporten la comunicació TWI o també anomenada i2c, utilitzant la llibreria corresponent.

5.3.4. Comunicació

L'Arduino té moltes facilitats de comunicació ja sigui amb un ordinador, un altre Arduino o amb altres Microcontroladors.

El ATmega328 proveeix de comunicació sèrie UART TTL, disponible a traves dels pins 0 i 1. El software de l'Arduino, inclou un monitor sèrie que permet enviar informació en forma de text cap a l'Arduino o al rebés. Hi ha dos leds a la placa, el RX i TX, que parpellegen quan sesta enviant informació cap a l'ordinador, mitjançant el port USB.

Una llibreria de Software es la que permet la comunicació sèrie en els pins digitals de l'Arduino

El Microcontrolador també suporta el I2C i les comunicacions SPI.

5.3.5. Programació

L'Arduino pot ser programat de varies formes, entre elles trobem:

Des de el propi programari d'Arduino, que pot ser programat des de les tres plataformes principals. Windows, Linux o MacOS.

O a traves del ICSP (In-Circuit Serial Programing) utilitzant el Arduino ISP o similar.

Per al meu cas, he considerat treballar amb el software Arduino IDE, ja que es més còmode, més intuïtiu i les llibreries existents són més amplies.

5.3.6. Protecció del USB per sobre tensió

L'Arduino disposa d'un fusible, que protegeix els ports USB de l'ordinador de curtcircuits i de intensitats massa elevades. Encara que la majoria d'ordinadors moderns ja disposen de protecció per a cassos com aquests, mai esta de més tenir una protecció extra.

Si s'apliquen més de 500 mA al port USB, el fusible automàticament tallarà la connexió fins que la sobrecàrrega es deixi d'aplicar.

5.3.7. Característiques físiques

La longitud de la placa Arduino Uno es de 6,86cm i 5,35cm d'amplada. Dels diferents elements que formen el conjunt, és el que ocupa més espai. és per això que s'ha substituït per L'Arduino Nano per tal de millorar l'espai del conjunt.

5.3.8. Resum

Les característiques que em vist de l'Arduino són les següents:

- El microcontrolador es el ATmega328.
- El voltatge operacional són 5V.
- El voltatge d'entrada hauria de ser d'entre 7V i 12V.
- El voltatge límit no pot baixar de 6V ni pujar de 20V.
- Entrades i sortides digitals 14, 6 de les quals tenen sortida PWM.
- Entrades analògiques 8.
- 40mA per als pins d'entrada i sortida.
- 50mA per al pin de 3,3V.
- Memòria Flash de 32KB, de les quals 0,5KB són pel bootloader
- Una SRAM de 2KB.
- EEPROM de 1KB.
- I velocitat de rellotge de 16MHz.

5.4. MEMÒRIA EEPROM

La memòria EEPROM, és un petit xip de silici utilitzat per a desar les dades de les diferents diferencies de pressió, que es calculen per saber si s'ha d'obrir el paracaigudes o no.

Les dades es desen des de que s'inicia el sistema, fins a que el dispositiu considera que ja ha acabat la seva funció. Com aquest moment es incert, i la memòria que porta incorporada l'Arduino es de tan sols 1KB, he decidit utilitzar una EEPROM externa que sigui de 32KB així puc emmagatzemar fins a 32 vegades més informació. Però anem a veure que suposa aquest canvi.

La memòria EEPROM utilitzada, consta de 32.000 posicions de memòria on cada una de les quals pot emmagatzemar un màxim de 256 bits o 8 bytes. Cada 100 mil·lèsimes de segon, el programa desa una dada dintre de la memòria EEPROM que tan sols ocupa 10 d'aquestes 100 mil·lèsimes de segon, en escriure aquesta informació en qualsevol dels seus registres de memòria.

Aleshores sabem que cada segon s'emmagatzemen 10 dades d'un total aproximat de 32.000. Si sabem que en una hora té 3600 segons, emmagatzemaríem 36.000 dades diferents cada hora.

Ara nomes hem de fer la divisió entre el total de dades per a saber els minuts que podrem estar emmagatzemant sense sobre escriure el contingut ja existent.

$$Nº\ d'hores = \frac{32.000}{36.000} = 0{,}89$$

Podrem estar emmagatzemant dades durant 53 minuts i 20 segons aproximadament.

5.4.1. Aspectes tècnics

Passem a veure aspectes més tècnics de la memòria EEPROM.

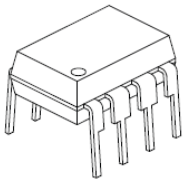
La memòria que he escollit és la memòria EEPROM 24LC32A, la qual disposa de 32Kbits de memòria. La comunicació amb L'Arduino, de la mateixa manera que passa amb el baròmetre, es fa utilitzant la comunicació sèrie I2C, que és ideal per a treballar amb Microcontroladors.

Algunes de les característiques d'aquest model son:

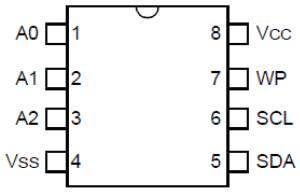
- El rang d'alimentació, oscil·la des dels 2,5v fins als 5,5v
- Pot suportar una freqüència de rellotge màxima de fins a 400KHz
- La velocitat d'escriptura màxima és de 5ms.
- Consta de protecció d'escriptura per software.
- Les dades poden ser esborrades o escrites més d'un milió de vegades.
- La retenció de les dades pot ser superior als 200 anys.
- Pot treballar des dels -40ºC fins als +85ºC.

5.4.2. Aspectes Físics

La memòria no és gaire gran ja que fa 0,92cm de longitud mentre que d'amplada nomes fa 0,63cm. Té unes dimensions estàndard i s'assimila al següent dibuix.



Consta de 4 potes per banda, les quals es descriuran a continuació:



A0, A1 i A2: són les potes numerades de la 1 a la 3. Aquestes, són entrades lògiques que poden prendre els valors de 0 ó 1 depenent de si es connecten a GND o a 5V. són utilitzades per saber la direcció física de la pròpia memòria, podent connectar fins a 7 memòries EEPROM diferents. Els valors que poden prendre serien des del 000 per al primer dispositiu, fins al 111 del vuite. Així sempre sabem a quina memòria EEPROM estem emmagatzemant les dades en el cas de que ni hagi més d'una.

VSS: Es la quarta pota i es el GND o alimentació negativa.

VCC: Es la vuitena pota i per la qual entraria l'alimentació positiva o els 5v.

WP: La setena pota es anomenada així per les sigles en angles de protecció contra escriptura(Write Protection). Quan se li aplica un corrent positiu de 5V, l'aparell no permet l'escriptura per a preservar les dades que conte, si per el contrari es connecta a GND o simplement no es connecta, la EEPROM permetrà que se li pugin escriure noves dades.

SCL: Es el clock del port I2C, utilitzat per a sincronitzar les dades. Es connecta al pin A5 del Arduino.

SDA: Un dels canals del bus I2C per on passen les dades, per anar i tornar. En el meu cas el connecto al pin A4 de l'Arduino.

6. SOFTWARE

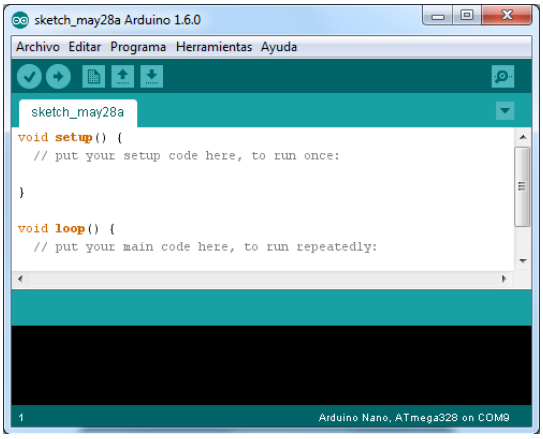
6.1. ARDUINO IDE

El codi de programació que porta el conjunt del paracaigudes, ha estat realitzat integrament utilitzant una eina de programació especialment dissenyada per a treballar sobre la plataforma Arduino. Aquesta eina s'anomena Arduino IDE (Integrated development environment), o en català entorn de desenvolupament integrat. Aquest entorn compta amb molts avantatges, tal com els següents:

- Es un entorn gratuït.
- Es un entorn lliure.
- Es multi plataforma, té suport per a Windows, Mac o inclús Linux.

A part, és bastant senzill de carregar el codi cap a l'Arduino. Si el codi compila correctament es qüestió de prémer un boto per a carregar el codi sobre la placa.

Com a punt negatiu destacaré que cal instal·lar el programa per a programar amb aquest. No es executable però la gran quantitat d'avantatges fan que sigui un punt menyspreable.



L'aspecte que presenta en Windows es el que mostra la imatge anterior.

Com ja he dit abans presenta moltes facilitats, com ara quan sobre una fulla de treball nova, escriu per pantalla la part on ha de anar el codi que nomes s'executarà una vegada i també la part del codi on s'anirà repetint tot el codi en bucle. Aleshores nomes es qüestió d'anar afegint el teu propi codi allà on pertoqui.

El programa permet programar totes les plaques Arduino següents:

Es versàtil i molt útil. Quan es tria la placa d'Arduino, ja esta preparada per a programar i el port en el qual es troba la placa el detecta automàticament.

Un altre avantatge que cal afegir al grup dels ja mencionats anteriorment, es el fet de que el propi programa ja porta exemples de codis fets.

Un cas particular. Si estàs utilitzant la EEPROM del propi microcontrolador, ja sigui per llegir o escriure sobre aquesta, té un apartat on mostra un programa d'exemple que explica com s'hauria de procedir per a realitzar aquesta tasca. El propi exemple incorpora també les llibreries que s'han d'utilitzar per al programa en qüestió, i cada apartat de codi esta comentat de tal manera que sigui un exemple clar i entenedor. Aleshores tu ja tens una idea de com has de procedir per a realitzar el teu propi programa.

Finalment, dir que per Internet hi ha gent que ofereix els seus programes o llibreries creades de forma altruista, per a que tothom les pugi utilitzar de forma gratuïta. Amb això vull dir que es un programa amb un gran suport darrere gracies als usuaris que l'utilitzen.

6.2. CODI DEL PARACAIGUDES

El llenguatge amb el que ha estat programat l'Arduino es el C++, encara que és possible programar-ho en altres llenguatges. No és un C++ pur sinó que és una barreja amb C.

El codi del paracaigudes, es compon d'un sol programa, però que es divideix en dos apartats principals que són els següents.

Aquests son:

- Apartat d'escriptura: Es l'encarregat de recollir les dades del sensor baromètric, analitzar-les i posteriorment emmagatzemar-les en la EEPROM externa. També decideix el moment d'obertura del paracaigudes en funció de les dades obtingudes, entre d'altres.

- Apartat de lectura: Aquest s'encarrega d'extreure les dades de la EEPROM, processar-les i finalment mostrar-les per pantalla.

Anem a veure el codi de cada un d'ells amb detall.

6.3. APARTAT D'ESCRIPTURA

Aquesta part esta pensada per a que la seva execució sigui mentre el conjunt estigui a l'aire. Si aquest programa falles, el paracaigudes no s'obriria i per tant tot el conjunt s'estavellaria contra el terra proporcionant resultats desastrosos.

Passem ara a veure el codi en si. Explicaré el codi per trossos per a facilitar la comprensió d'aquest.

6.3.1. Llibreries

Anomenaré les llibreries que utilitza aquest apartat del codi.

```
#include <SFE_BMP180.h>
#include <Wire.h>
#include <Servo.h>
```

Aquestes llibreries fan les següents funcions:

- SFE_BMP180: És la llibreria utilitzada pel baròmetre, ha estat creada especialment per al baròmetre utilitzat. Aquesta es pot trobar a la pròpia pagina del fabricant del baròmetre i esta sota llicencia beerware.
- Wire: Aquesta llibreria es la que s'utilitza per a poder establir la comunicació i2c entre els diferents dispositius a connectar mitjançant aquest protocol de comunicació. Els dispositius que utilitzaran aquest protocol seran: L'Arduino, el sensor baromètric i finalment la memòria EEPROM. Aquesta llibreria ja ve inclosa dintre de l'entorn de programació Arduino IDE.
- Servo: És la tercera llibreria i s'utilitza pel servo motor. Ens permet determinar la posició en la qual es trobarà el motor i també assignar el senyal del motor a una de les potes de l'Arduino. Aquesta llibreria també ve inclosa dintre de l'entorn de programació Arduino IDE.

6.3.2. Variables i definicions

Les variables que he utilitzat són les següents.

```
#define disk1 0x50

Servo myservo;

SFE_BMP180 pressure;

double presio, pres_ant, pres_res, mitj_pres, mitj_ant, baseline, altura, ober, desa;
```

```
int pos = 0, x = 90, a = 0, obertura = 0, lect_escr = 0, compt_fi = 0;

char c;

unsigned int address = 0, alt_baix, alt_fi, tura;
```

- La primeria línea és on es defineix que el disk1 es troba a la posició 0x50. Se li dona un nom per a fer més fàcil el fet de treballar amb la EEPROM, ja que se'n podrien posar fins a 8 diferents.
- A la segona i tercera línia definim el nom del servomotor i el del sensor baromètric respectivament.
- Seguidament definim les següents variables com a *double*, ja que contenen nombres amb decimals:
    - **presio**: És la variable on hi guardarem la pressió actual recollida pel sensor.
    - **pres_ant**: És allà on guardarem la pressió anterior, seguidament ho explicaré amb més detall.
    - **pres_res**: És on es guardarà la diferencia de pressions, és també la variable que s'utilitza per decidir si sobre el paracaigudes o no, depenen de la grandària d'aquesta.
    - **mitj_pres**: Si van sumant les diferents pressions preses al inici i es divideix entre el nombre de pressions, per a fer una mitjana i el valor sigui més estable
    - **mitj_ant**: Usada per a desar els valors de les mitjanes anteriors.
    - **baseline**: S'utilitza per a determinar l'altura relativa a la que sobre el paracaigudes respecte de la posició d'inici. Es desa la pressió inicial.
    - **altura**: Es desa la pressió d'obertura del servomotor per a determinar l'altura.
    - **ober**: S'hi desa el valor que ha activat el servomotor.
    - **desa**: S'utilitza per guardar el valor de la variable **mitj_ant** durant l'execució del codi.
- Les següents variables són les definides com a *int*:
    - **pos i x**: són les posicions a les que el servomotor es posicionarà depenent de les diferents parts del codi.
    - **a**: Es la variable on es guarda l'altura d'obertura del paracaigudes, respecte del terra.
    - **obertura**: El seu valor varia a 1 quan el paracaigudes s'ha obert.
    - **lect_escr**: Es la variable que ajuda en l'elecció del tros de codi que s'executarà.
    - **compt_fi**: Ajuda a determinar quan s'ha acabat el programa.
- Per als tipus char nomes tenim:
    - **c**: Es la variable on es desarà alguna cosa en cas de que arribi pel port serie.
- Finalment per al unsigned int:
    - **address**: Es la posició de memòria on es desaran els diferents valors a la EEPROM.
    - **alt_baix**: Correspon a la posició de memòria de l'últim valor de la baixada.
    - **alt_fi**: És la posició de memòria de l'últim valor útil de la EEPROM.
    - **tura**: Si desa la posició de memòria del valor d'obertura.

6.3.3. Setup

La següent part del codi és la que es correspon al 'void setup()' o al codi que nomes es repeteix una vegada en l'execució del programa. El codi descrit a continuació es el que va dintre d'aquest apartat.

El primer que trobem es

```
Serial.begin(9600);
Wire.begin();
```

Inicialitzem el port serial i també el protocol de comunicació i2c, per establir la connexió entre els diversos dispositius.

```
Serial.println ("Introdueix qualsevol tecla per inicialitzar programa de lectura.");
Serial.println ("Esperant tecla...");

delay(5000);
Serial.print ("5..");
delay(1000);
Serial.print ("4..");
delay(1000);
Serial.print ("3..");
delay(1000);
Serial.print ("2..");
delay(1000);
Serial.println ("1");
delay(1000);
```

El següent que trobem, és un petit menú que es mostrarà en el cas que entrem en el monitor sèrie que incorpora el programa. I un compte enrere amb un total de 10 segons per a que a l'usuari li doni temps d'entrar una dada en el cas que es vulgui entrar al programa de lectura.

```
Serial.println ("Entrant al programa d'escriptura");
lect_escr = 2;

myservo.attach(6);
myservo.write(pos);
```

Si no s'ha rebut res pel port sèrie, s'entra al programa d'escriptura, i a lect_escr se li assigna un dos per a que posteriorment executi aquesta part. A continuació es lliga el comportament del servo motor a la pota 6 de l'Arduino. És a dir per on s'enviarà la senyal de posició a la qual es tindrà que col·locar aquest. I tambe s'indica al servomotor la posició de tancament o la posició inicial que ha de tenir per a que no s'alliberi el paracaigudes. El servo motor romandrà en aquesta posició fins que es tingui que alliberar el paracaigudes.

```
if (pressure.begin())
    Serial.println("Baròmetre funcionant!");

  else
  {
    Serial.println("Quelcom a fallat, surt i torna a entrar.");
    while(1);
  }
```

Posem una condició lògica per veure si funciona correctament el sensor baromètric. Si aquest funciona correctament sens mostra, a traves del port serial un missatge de que tot esta anant

be. Si pel contrari el sensor no estigues ben connectat o falles alguna cosa, sens mostraria un missatge d'error pel port sèrie, i l'execució del programa s'aturaria en aquest punt fins que es reinicies el programa.

```
pres_ant = getPressure();
presio = pres_ant;
baseline = pres_ant;
mitj_pres = getPressure();
baseline = mitj_pres + baseline
mitj_pres = getPressure();
baseline = (mitj_pres + baseline)/3;
mitj_ant = baseline;
```

Finalment, es pren la pressió atmosfèrica per a tenir un valor de referència, i es desa en 3 variables diferents: **pres_ant, presio i baseline**. No obstant **baseline** es la mitjana de tres valors per a poder determinar millor l'altura. A la variable **mitj_ant** si desa el valor d'aquesta mitjana, per a tenir un valor per comparar per primer cop.

6.3.4. Loop

Aquesta és la part del codi que es va repetint indefinidament fins que es compleixi un esdeveniment determinat.

```
if(lect_escr == 2)
```

Com que **lect_escr** a passat a ser 2 entrem en l'apartat d'escriptura.

```
digitalWrite(13, LOW);
delay(58);
```

Comença amb el led parat i esperem 58ms per a començar el programa en si.

```
presio = getPressure();     // 1a mostra
mitj_pres = presio;
pres_res = (presio - pres_ant) * 100;
writeEEPROM(disk1, address, pres_res); //= 10 ms d'espera
address++;
pres_ant = presio;
delay(58);

presio = getPressure();     // 2na mostra
mitj_pres = presio + mitj_pres;
pres_res = (presio - pres_ant) * 100;
writeEEPROM(disk1, address, pres_res); //= 10 ms d'espera
address++;
pres_ant = presio;
delay(58);

presio = getPressure();     // 3a mostra
mitj_pres = presio + mitj_pres;
pres_res = (presio - pres_ant) * 100;
writeEEPROM(disk1, address, pres_res); //= 10 ms d'espera
address++;
pres_ant = presio;
delay(58);

presio = getPressure();     // 4a mostra
mitj_pres = presio + mitj_pres;
pres_res = (presio - pres_ant) * 100;
writeEEPROM(disk1, address, pres_res); //= 10 ms d'espera
```

```
address++;
pres_ant = presio;
delay(58);

presio = getPressure();     // 5a mostra
mitj_pres = (presio + mitj_pres)/5;
pres_res = (presio - pres_ant) * 100;
writeEEPROM(disk1, address, pres_res); //= 10 ms d'espera
pres_ant = presio;
```

Es prenen 5 mostres diferents amb una diferencia de 100ms entre cada una d'elles. Cada un de les mostres es desa a la seva posició corresponent de la memòria EEPROM.

Després es fa la mitjana de les 5 mostres per a obtenir un sol valor més acurat, que s'utilitzarà per fer la diferencia, i determinar si obrir o no el paracaigudes.

```
pres_res = (mitj_pres - mitj_ant) * 100;
Serial.println(pres_res);
desa = mitj_pres;
digitalWrite(13, HIGH);
```

El següent que succeeix, es la resta entre la pressió presa actualment i l'anterior. Aquesta part es important ja que es la que determinarà si el paracaigudes s'ha d'obrir o no. Aquesta resta s'ha de multiplicar per 100, ja que la diferencia entre una pressió i altre en el espai de temps de 500ms es de centèsimes, i per tal de poder desar la informació a la EEPROM es necessari.

La resta funciona de la següent manera:

- Si el resultat d'aquesta és negatiu, ens indicarà que el conjunt del paracaigudes esta pujant degut a que la variable **presio** serà menor a la **pres_ant** ja que com s'ha explicat en l'apartat del baròmetre, a major altura menor pressió atmosfèrica.

- Si per el contrari, el resultat de la resta es positiu significarà que el conjunt del paracaigudes estarà en descens per la mateixa raó explicada anteriorment. Depenent de la grandària d'aquesta resta, es decideix si obrir o no el paracaigudes.

Seguidament s'envia el resultat de la resta pel port sèrie per a poder-lo visualitzar amb immediatesa en un monitor. No obstant aquest valor no es desa a la memòria EEPROM. El fet de que el resultat es mostri pel port sèrie és degut a les proves que san realitzat, per a veure el correcte funcionament del codi.

També es desa el valor de la mitjana en una variable per a utilitzar-la en la següent comprovació.

Finalment s'encén el LED propi de l'Arduino per a indicar que el programa esta en execució.

```
if(pres_res>=13 && obertura == 0)
  {
    address++;
    presio = getPressure();
    pres_res = (presio - pres_ant) * 100;
    writeEEPROM(disk1, address, pres_res);
    Serial.print("Pressio > 10: ");
    Serial.println(pres_res);
```

Entrem en la primera condició lògica per a determinar si el paracaigudes esta en descens o no. Comparem el resultat obtingut de la resta amb el valor 13. Tan aquest valor com el següent que es 20, s'han determinat a partir de les proves realitzades.

Si el valor de **pres_res** es menor, no entra dintre i continua fora de la condició lògica. Si resulta que el valor és més gran o igual a 13, sumem una posició a la adreça de la EEPROM, es torna a mesurar la pressió per a fer un nou càlcul, es realitza la resta, es desa el resultat en una nova posició de memòria de la EEPROM, i finalment es mostra per pantalla.

```
if(pres_res>=20 && obertura == 0)
    {
      myservo.write(x);
      Serial.println("Paracaigudes obert");

      ober = pres_res;
      tura = address;

      altura = getPressure();
      mitj_pres = altura;
      altura = getPressure();
      mitj_pres = mitj_pres + altura;
      altura = getPressure();
      altura = (mitj_pres + altura)/3;

      obertura=1;
    }
```

Es torna a fer una comprovació, amb el resultat obtingut de la resta anterior, per evitar falsos positius. Si aquest valor és menor a 20, no entrarà dins el **if** i continuarà amb el codi. Si pel contrari entra dins, significa que realment el dispositiu esta caient i immediatament es dona el senyal al motor per a que es posi en la posició d'obertura i desplegui el paracaigudes. Abans d'aturar el programa però es fan un seguit d'accions per a saber que ha passat:

- El primer que es fa, avisar pel port sèrie de que el compartiment del paracaigudes s'ha obert.
- També es desen en variables diferents el valor d'obertura i l'adreça de memòria d'aquest valor. Es desen en variables per a després escriure-les més endavant a la memòria EEPROM.
- Seguidament, s'agafa la pressió atmosfèrica 3 vegades i es fa la mitjana d'aquesta pressió. Es desa en la variable **altura**, per a posteriorment determinar l'altura d'obertura.
- finalment s'assigna 1 a la variable **obertura** per evitar que es torni a entrar en aquest apartat del codi.

després de que el paracaigudes sigui desplegat, podrem accedir a la següent part del codi

```
if (obertura == 1)
  {
    if (1 > pres_res && pres_res > -1)
       {
         compt_fi++;
         if(compt_fi == 3)
         {
         alt_baix = address;
         }

         if(compt_fi == 10)
         {
```

```
        alt_fi = address;
```

Aquesta es la part final i nomes si pot accedir un cop el paracaigudes estigui desplegat. La manera que tinc per a determinar que el dispositiu esta aturat es comptant fins a 10 valors petits en la resta entre pressions. Abans però, per a determinar el final de la baixada es desarà el valor final d'aquesta quan rebi 3 d'aquests valors. Mentre no s'entri en el if que determina que el programa ja s'ha acabat, el codi que s'executarà des del inici, però sense poder entrar a l'estat alarma ni al estat obertura, ja que el paracaigudes ja es trobarà desplegat.

Quan s'ha determinat que l'objecte esta parat, s'entra en la última part del codi.

```
address++;
writeEEPROM(disk1, address, -120);
address++;
writeEEPROM(disk1, address, -120);
address++;
writeEEPROM(disk1, address, -120);
a = pressure.altitude(altura,baseline);

Serial.print("Altura d'obertura respecte del terra: ");
Serial.print(a,1);
Serial.println(" metres.");

address++;
writeEEPROM(disk1, address, a);
a = a/256;
address++;
writeEEPROM(disk1, address, a);
```

El que succeeix es el següent:

- Primerament, s'escriuen els tres sentinelles consecutius per a determinar fins a on ha arribat l'execució del programa.
- A les hores es determina l'altura en funció del contingut de les variables **baseline** i **altura**. Que són les pressions inicial i d'obertura del paracaigudes respectivament. Aquesta altura també s'envia pel port serial.
- Un cop tenim l'altura s'ha de desar, però aquí ens apareix un problema degut a les limitacions de la EEPROM. El problema és que a cada posició de memòria nomes i cap un nombre compres entre 0 i 255, i si per exemple es volgués guardar el nombre 3000, dons no hi cabria.

La solució trobada a aquest problema és la següent:

Imaginem que efectivament el paracaigudes s'ha obert a 3000 metres d'alçada, i desem aquest valor dintre de la posició de memòria després del sentinella, el valor desat serà 184.

$$3000 = 184 + 256 \cdot 11$$

Aquest 184 es el residu de la divisió entera entre 3000 i 256. Aleshores en la següent posició de memòria es desa el quocient, que en aquest cas es el numero 11.

-120
-120
-120

184
11

Les ultimes posicions de memòria de la EEPROM ens quedaran tal com es veu a la taula, i així amb el 184 i el 11 podrem obtenir l'altura a la que s'ha desplegat el paracaigudes.

```
address++;
writeEEPROM(disk1, address, ober);
address++;
writeEEPROM(disk1, address, tura);
tura = tura/256;
address++;
writeEEPROM(disk1, address, tura);

address++;
writeEEPROM(disk1, address, alt_baix);
alt_baix = alt_baix/256;
address++;
writeEEPROM(disk1, address, alt_baix);

address++;
writeEEPROM(disk1, address, alt_fi
alt_fi = alt_fi/256;
address++;
writeEEPROM(disk1, address, alt_fi);

digitalWrite(13, LOW);
while(1);
```

Després de l'altura es desarà el valor de l'obertura i també la posició d'aquesta en la memòria EEPROM. Es segueix el mateix procediment que per desar l'altura i es desa en dos parts, ja que el valor de la posició de la memòria pot ser superior a 255. A continuació també es desen de la mateixa forma les posicions per al final de la baixada i l'últim valor de tots. Aquestes posicions, nomes es desen amb la finalitat de facilitar la lectura de la gràfica que es crearà a partir del valors obtinguts. Amb les posicions de memòria es crearà una llegenda per aquesta gràfica que n'indicarà els diferents estats.

Un cop hem desat els valors podem aturar el programa amb el **while(1),** abans però parant el led.

```
mitj_ant = desa;
pres_ant = presio;
address++;
```

En el cas de que no agüéssim entrat en el segon **if**, aquestes tres línies de codi s'agüessin executat, desant la mitjana de presions, tambe l'última pressió presa com a **pres_ant**, i augmentant en una la posició de l'adreça de la memòria EEPROM.

Després el codi es va executant en bucle fins que no entri en el segon if i s'aturi el programa.

6.3.5. Get Pressure

Aquesta funció no ha estat creada per mi sinó per algú sota llicencia beerware, fa mesures de la temperatura per a calcular la pressió ambient i retorna el resultat per a que pugi treballar amb aquesta.

6.3.6. Write EEPROM

Es la funció encarregada d'escriure en la posició corresponent de la memòria, rep tres paràmetres; La direcció de la memòria EEPROM, la posició de memòria on ha de escriure en la EEPROM, i finalment la informació a escriure dintre d'aquesta posició.

Els passos que segueix són els següents:

- Inicia la comunicació i2c amb el dispositiu corresponent.
  - `Wire.beginTransmission(deviceaddress);`
- Escriu el bit més i menys significatiu respectivament de l'adreça.
  - `Wire.write((int)(eeaddress >> 8));  // MSB`
  - `Wire.write((int)(eeaddress & 0xFF)); // LSB`
- Aleshores s'escriu la informació que es vol desar a dins de la EEPROM.
  - `Wire.write(data);`
- Finalment es tenca la comunicació i2c i s'espera 10ms que es el temps que triga en escriure la informació en la memòria EEPROM.
  - `Wire.endTransmission();`
  - `delay(10);`

6.3.7. Exemple d'escriptura

El que es veu a continuació es un exemple de l'execució del programa anterior i vist des del port sèrie.

```
Introdueix qualsevol tecla per inicialitzar programa de lectura.
Esperant tecla...
5..4..3..2..1
Entrant al programa d'escriptura.
Barometre funcionant!
4.02
0.03
-0.60
-0.24
-2.06
25.18
Pressio >= 13: 29.62
Paracaigudes obert
-2.09
-3.12
-8.53
-1.00
3.42
-0.37
3.95
-1.97
-0.67
0.34
0.60
-0.16
1.89
0.54
0.20
```

```
0.06
-2.64
-0.18
2.13
0.57
Altura d'obertura respecte del terra: 0 metres.
```

Aquest petit experiment tan sols a durat uns segons, però recull una bona quantitat de dades.

Quan entrem en un monitor de port sèrie, el primer que ens envia L'Arduino, es un petit missatge on diu que si introduïm qualsevol tecla entrarem en l'apartat de lectura de les dades de la memòria EEPROM. Com que no es el cas, després d'esperar 10 segons, automàticament entrarem en l'apartat d'escriptura. On rebrem els valors del sensor baromètric i es mostraran pel port sèrie tal i com es pot apreciar.

Es calcula la diferencia de pressió entre la mitjana de les 5 mostres actuals i les 5 anteriors. Aquesta diferencia serà el valor que es mostra per pantalla i que es decisiu, per tal de decidir el moment d'obertura del paracaigudes. No obstant cada una de les 5 mostres que formen la mitjana, es desen a la memòria EEPROM per separat per a després poder realitzar el gràfic de tot el conjunt de les dades.

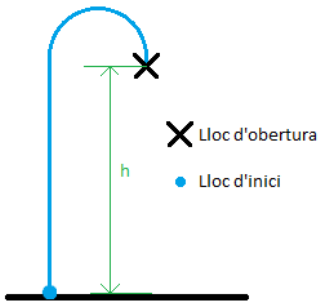
Quan els valors oscil·len entre 4 i -4, es degut a que l'aparell no esta en moviment o no té un moviment acusat. En altres paraules, que no es mou a grans velocitats.

Si els valors són majors de 5 o menors de -5 vol dir que l'aparell esta baixant o pujant a certa velocitat.

Quan el valor ja es superior a 13 clarament estem baixant, com es el nostre cas on rebem dos valors grans consecutius 25 i 29. es a les hores quan es mostra per pantalla que el paracaigudes s'ha obert.

Com es pot observar els últims valors són força petits això ens indica que el paracaigudes a arribat al final del seu recorregut i esta aturat. En aquest punt el programa decideix aturar-se, col·locar els sentinelles al final de les dades capturades i mostrar per pantalla l'altura d'obertura respecte del terra. En aquest cas el valor que marca es de 0 metres degut a que s'ha obert a aquesta alçada.

Però el programa esta pensat per a que el paracaigudes s'obri a grans altures.



Un exemple de comportament seria aquest. El dispositiu pren la primera mesura en el lloc d'inici, seguidament inicia el seu vol en vertical, i acaba entrant en caiguda lliure. Quan es detecta que esta en caiguda lliure, s'obre el paracaigudes i aleshores es pren l'última mesura.

A partir de la primera i última pressió atmosfèrica presa, es calcula l'altura a la que s'ha obert el paracaigudes. Aquesta altura es mostra per pantalla i també es desa a la EEPROM tal com s'ha explicat.

Després de l'altura també es desen els valors que ens serviran per a entendre millor la gràfica. però que no queden reflectits en el port sèrie.

6.4. APARTAT DE LECTURA

La part de lectura, és l'encarregada d'extreure les dades emmagatzemades de l'interior de la memòria EEPROM, corresponents a l'última sessió enregistrada. Acabada de llegir la dada s'extreu pel port sèrie i es mostra per pantalla.

Cal dir que aquest programa a diferencia del principal, esta pensat per a que s'executi en un entorn controlat per tant els temps d'execució no són crucials.

6.4.1. Llibreries

L'única llibreria que utilitza aquesta part, és la llibreria Wire.

```
#include <Wire.h>
```

Aquesta llibreria fa la següent funció:

- Wire: Aquesta llibreria és la que s'utilitza per a poder establir la comunicació i2c entre els dos dispositius a connectar mitjançant aquest protocol de comunicació. Els dispositius que utilitzaran aquest protocol seran: L'Arduino i la memòria EEPROM. Aquesta llibreria ja ve inclosa dintre de l'entorn de programació Arduino IDE.

6.4.2. Variables i definicions

```
#define disk1 0x50

int lect_escr = 0;

uint8_t alt, alt2, altu;

int8_t ver = 0, val;

unsigned int address = 0,altura_rel;
```

La definició del disk1 tan sols es per a facilitar la feina per l'adreçament de la memòria EEPROM.

Les variables són les següents:

- **lect_escr**: Quan es rep quelcom pel port sèrie es posa a 1, aleshores s'executa el programa de lectura.
- **alt**: Es on es desarà un dels dos valors per a formar l'altura de desplegament del paracaigudes, també si desen la primera part d'algunes posicions de memòria com ja he explicat. Es un integer sense signe de 8 bits, ja que el valors nomes poden ser positius.
- **alt2**: Es el valor que s'utilitza junt amb Alt per a saber l'altura d'obertura, i la segona part. Es un valor enter sense signe de 8 bits, ja que els valors nomes poden ser positius.
- **altu**: Variable on es desa el valor que ha fet desplegar el paracaigudes.
- **ver**: La variable utilitzada per a desar cada un dels resultats extrets de la memòria EEPROM, inclosos els sentinelles. Es un integer amb signe de 8 bits, ja que els valors poden ser tan positius com negatius.
- **val**: Determina el nombre de sentinelles llegit.
- **address:** Es correspon a la variable que determina la posició d'on es llegirà la memòria EEPROM. Es un enter sense signe perquè va de 0 fins a 32k posicions.
- **altura_rel**: Es la variable on es desa el resultat calculat a partir de l'altura d'inici i l'altura d'obertura del paracaigudes. Es un integer sense signe, degut a que l'altura d'obertura sempre hauria de ser major a la de d'inici.

6.4.3. Setup

El codi que nomes s'executa una sola vegada és el següent.

```
Serial.begin(9600);
Wire.begin();
```

Aquesta part de codi es compartida amb l'apartat d'escriptura, on.

- S'inicialitza el port sèrie, amb un baud rate de 9600.
- i s'inicia la comunicació i2c.

```
if (Serial.available() > 0)
{
   lect_escr = 1;
   Serial.println ("Entrant al programa de lectura");
   delay(1000);
}
```

Després del menú vist en l'apartat d'escriptura que és compartit amb el de lectura ens trobem amb la següent condició, on si hem rebut alguna cosa pel port sèrie hi entrara.

Aleshores la variable **lec_escr** rep el valor d'1 per a posterior ment entrar en l'apartat de lectura dintre del codi cíclic. S'envia el missatge com que s'entra en el programa de lectura. i s'espera 1 segon per continuar.

6.4.4. Loop

Anem a analitzar el codi que es repeteix en bucle. En aquest apartat del codi nomes si pot entrar quan el valor de **lect_escr** es de 1.

```
else if(lect_escr == 1){
val = 0;
ver = readEEPROM(disk1, address);
if (ver >= 0.00) Serial.print(" ");
Serial.println(ver);
if (ver == -120)
```

Els passos que segueix són els següents:

- Primerament inicialitzem **val** a 0.
- A continuació llegim la primera posició de la memòria EEPROM i la desem a al variable **ver**.
- Seguidament fem una petita comprovació tan sols per mirar si el valor es positiu o no, i afegir un espai per a que es mostri alineat pel port sèrie en el cas de que el valor sigui positiu.
- S'agafa el valor desat en la variable **ver** i es mostra per pantalla.
- Per finalitzar, es comprova si el valor de la variable es un dels sentinelles o no. En el cas de que ho sigui, s'entrarà dintre de la condició lògica per a comprovar si el següent també o és. En el cas de que no ho sigui, es sumarà a l'adreça de la posició de memòria una posició i es tornarà a executar aquesta part del programa.

```
address++;
val++;
ver = readEEPROM(disk1, address);
if (ver >= 0.00) Serial.print(" ");
Serial.println(ver);
if (ver == -120)
```

Un cop dins del **if** ens trobem amb les següents línies de codi. Són exactament les mateixes que acabem de veure, a excepció de la variable **address** que augmenta en una posició per evitar tornar a llegir les mateixes dades, i la variable **val** fa el mateix indicant així que hem trobat un sentinella.

A continuació, es tornar a llegir de la EEPROM, emmagatzemar el valor en **ver**, mostrar-lo per pantalla i finalment comprovar si aquest nou valor es un sentinella (-120) o no. Si no ho és sortirem del **if** i continuarem en el bucle del codi. Si pel contrari si que ho és, entrarem dins del proper **if**.

El tros de codi que segueix a l'últim vist, és exactament el mateix.

```
address++;
val++;
ver = readEEPROM(disk1, address);
if (ver >= 0.00) Serial.print(" ");
Serial.println(ver);
if (ver == -120)
{
        val++;
        if (val == 3)
```

Es una mica repetitiu però serveix per a comprovar si els tres sentinelles, el valor -120, estan seguits en el codi. Quan es troben els 3 sentinelles seguits aleshores s'entra dins del ultim if, significant que no hi ha més dades a excempcio de les que venen després dels sentinelles.

```
address++;
alt = readEEPROM(disk1, address);
Serial.println(alt);
address++;
alt2 = readEEPROM(disk1, address);
Serial.println(alt2);
altura_rel = alt + (alt2 * 256);
Serial.print("L'altura d'obertura respecte del terra es de ");
Serial.print(altura_rel);
Serial.println(" metres.");
```

Quan estem en aquesta part de codi, significa que la part de dades capturades ja s'ha acabat o gaire bé. Ja que després dels sentinelles es troba l'altura a la que s'ha desplegat el paracaigudes respecte del terra i també les diferents posicions de memòria que indiquen les diferents fases de l'execució del programa.

El codi fa el següent:

- S'afegeix una posició més a l'adreça. Seguidament es llegeix el valor següent al sentinella que es la primera part de l'altura, i es desa en **alt**.
- S'envia pel port sèrie i s'incrementa de nou l'adreça. No faria falta visualitzar aquesta dada, però serveix per comprovar si hi ha algun error.
- Es torna a llegir la memòria EEPROM, s'emmagatzema en **alt2** i seguidament es mostra per pantalla a traves del port sèrie. De la mateixa manera que la variable alt nomes ens serveix com a comprovació.
- Es fan els càlculs per a determinar l'altura a la que s'ha desplegat el paracaigudes. Com ja s'ha dit, cada posició de memòria nomes pot emmagatzemar fins a 256 valors diferents començant per 0 i acabant per 255. Es per això que si l'altura fos de 300 m no la veuríem realment i es necessita de 2 posicions de memòria per a fer els càlculs.
    - Suposem que l'altura d'obertura han estat 500 metres ,aleshores **alt** = 244 i **alt2** = 1

$$altura_{rel} = 244 + 1 * 256 = 500$$

- El valor de l'altura es desa en altura_rel i s'envia pel port sèrie.

A continuació seguim amb els ultims valors introduits a la EEPROM.

```
address++;
altu = readEEPROM(disk1, address);
address++;
alt = readEEPROM(disk1, address);
address++;
alt2 = readEEPROM(disk1, address);
altura_rel = 1 + alt + (alt2 * 256);

Serial.println("Llegenda de la gràfica: ");
Serial.println("Es descriu la posició dels valors per interpretar la gràfica.");

Serial.print("Inici ( 1 - ");
Serial.print(altura_rel - 2);
```

```
Serial.println(" )");

Serial.print("Alarma ( ");
Serial.print(altura_rel - 1);
Serial.println(" )");

Serial.print("Obertura ( ");
Serial.print(altura_rel);
Serial.print(" ) amb valor ");
Serial.println(altu);

Serial.print("Baixada ( ");
Serial.print(altura_rel + 1);
Serial.print(" - ");

address++;
alt = readEEPROM(disk1, address);
address++;
alt2 = readEEPROM(disk1, address);
altura_rel = 1 + alt + (alt2 * 256);

Serial.print(altura_rel);
Serial.println(" )");

Serial.print("Aturada ( ");
Serial.print(altura_rel + 1);
Serial.print(" - ");

address++;
alt = readEEPROM(disk1, address);
address++;
alt2 = readEEPROM(disk1, address);
altura_rel = 1 + alt + (alt2 * 256);
Serial.print(altura_rel);
Serial.println(" )");

while (1);
```

Aquesta part del codi ha estat creada bàsicament per entendre la gràfica que generaran les dades. Es tracta d'una llegenda per a poder-la llegir correctament.

Primerament en la variable **altu** es desa el valor que ha fet obrir el paracaigudes. I en **altura_rel** la posició de memòria que ocupa aquesta dada.

La llegenda consta de cinc estats diferents que son:

- **Inici**: Va des de la primera dada enregistrada, fins a la anterior a la alarma. S'utilitza la variable **altura_rel** i se li resta dos posicions per tal de determinar el final d'aquest estat.
- **Alarma**: Es l'estat posterior a inici, i el que conte el valor anterior al que farà saltar el paracaigudes. La posició de la memòria on es troba el valor de alarma es troba a partir de restar un valor a altura_rel.
- **Obertura**: Com el seu nom indica mostra el valor d'obertura i la posició que ocupa aquest. El valor es troba a la variable **altu** i la posició que ocupa a la memòria en **altura_rel**.
- **Baixada**: Es tracta del estat que descriu la baixada del conjunt després de l'obertura. Per al primer valor d'aquest estat utilitzem el valor de la posició de memòria anterior sumant una posició. Per a indicar el final d'aquest estat hem de tornar a llegir de la memòria EEPROM ja que esta guardat en les següents posicions. El final de la baixada

es determina quan san rebut tres valors suficientment petits com per a determinar que el conjunt ja no esta baixant.

- **Aturada**: Es l'estat final dels cinc i si accedeix quan el dispositiu roman aturat. Per a determinar l'inici d'aquest estat, sumem una posició al final de l'estat anterior. Mentre que pera determinar-ne el final ,sa de llegir de la memòria EEPROM per a saber quina ha estat l'última posició útil escrita en aquesta.

Després de la llegenda, el programa s'atura en el while(1).

6.4.5. Read EEPROM

Es l'última part d'aquest codi, i la funció encarregada de retornar el contingut de la memòria EEPROM.

```
int8_t readEEPROM(int deviceaddress, unsigned int eeaddress )          //llegir eeprom
{
  int8_t rdata;

  Wire.beginTransmission(deviceaddress);
  Wire.write((int)(eeaddress >> 8));   // MSB
  Wire.write((int)(eeaddress & 0xFF)); // LSB
  Wire.endTransmission();
  Wire.requestFrom(deviceaddress,1);

  if (Wire.available()) rdata = Wire.read();

  return rdata;
}
```

A diferencia del writeEEPROM, aquesta funció no rep tres paràmetres sinó dos. Nomes rep la direcció del la memòria EEPROM i la direcció on ha de llegir dintre d'aquesta. Aleshores si tot es correcte es retorna la informació continguda dintre de la memòria EEPROM en format de 8 bits sense signar, es a dir que va des de -127 fins a 128, ja que alguns valors són negatius.

6.4.6. Exemple del programa

```
L'altura d'obertura respecte del terra es de 0 metres.
Introdueix qualsevol tecla per inicialitzar programa de lectura.
Esperant tecla...
5..4..3..2..1
Entrant al programa de lectura
 0
 0
-1
 1
 1
-2
 1
-1
 2
-1
 0
 0
 0
-3
-1
-3
-1
-5
-3
```

```
 0
-2
 0
-1
 0
 1
-2
 3
 1
-1
-5
-1
-7
-1
-4
 0
 0
 0
 13
 30
-5
 0
-2
-1
-3
-3
 3
 0
 0
 1
 1
-1
 0
 1
 0
 0
 0
 3
 0
 1
 0
 0
-120
-120
-120
0
0
L'altura d'obertura respecte del terra es de 0 metres.
Llegenda de la gràfica:
Es descriu la posició dels valors per interpretar la gràfica.
Inici ( 1 - 37 )
Alarma ( 38 )
Obertura ( 39 ) amb valor 30
Baixada ( 40 - 48 )
Aturada ( 49 - 61 )
```

Al visualitzar les dades pel port sèrie, primerament veiem el mateix que en l'apartat d'escriptura, però al enviar qualsevol caràcter per aquest port i després de 10 segons, el programa de lectura s'iniciarà.

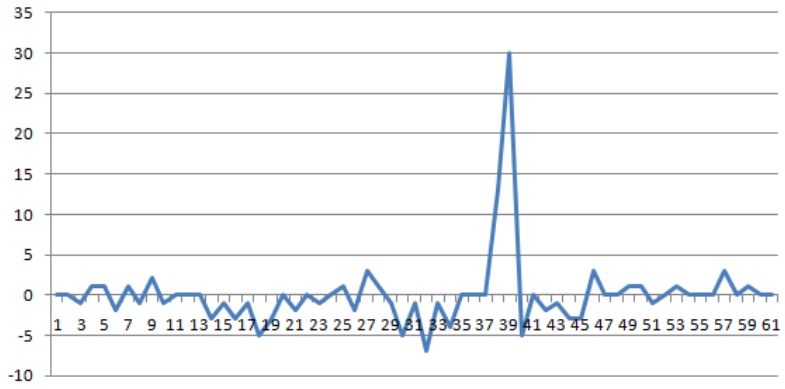
Com Podem apreciar els valors obtinguts són els mateixos que els resultat de l'exemple d'escriptura, però variant nomes la manera de visualitzar-los. Els valors es mostren sense decimals i un rere l'altre.

Aquí no es mostra cap missatge per pantalla dient que el paracaigudes s'ha obert però es pot apreciar clarament en els valors 13 i 30 que ha estat alla. Això es fa d'aquesta manera per a poder agafar totes les dades útils i poder-les representar en un gràfic.

després veiem que els valors següents són relativament petits i quan el programa en té uns quants decideix aturar-se.

cal dir que entre valor i valor hi ha una diferencia de 150ms, per tant cada segon es desen 6 dades en la memoria eprom. Aixi podem saber el total de temps transcorregut en funció de les dades que tinguem. amb les 61 dades que tenim podem saber que el temps transcoregut es de 10 segons aproximadament.

El gràfic obtingut seria el següent:



Si ens fixem en el gràfic veiem que:

- L'eix d'abscisses es el temps transcorregut.
- I el d'ordenades es la diferencia de pressió entre mesures, es a dir els valors desats a la memòria EEPROM.

Si observem la llegenda:

```
Inici ( 1 - 37 )
Alarma ( 38 )
Obertura ( 39 ) amb valor 30
Baixada ( 40 - 48 )
Aturada ( 49 - 61 )
```

Podem apreciar l'estat Inici es correspon a les dades estables del principi, on els valors no són més grans de 5 o inferiors a -5 amb una exempció que es correspon a la pujada de l'aparell casi al final d'aquest estat.

L'estat Alarma nomes disposa d'un sol valor i es correspon al valor que detecta que l'aparell ha començat a descendir.

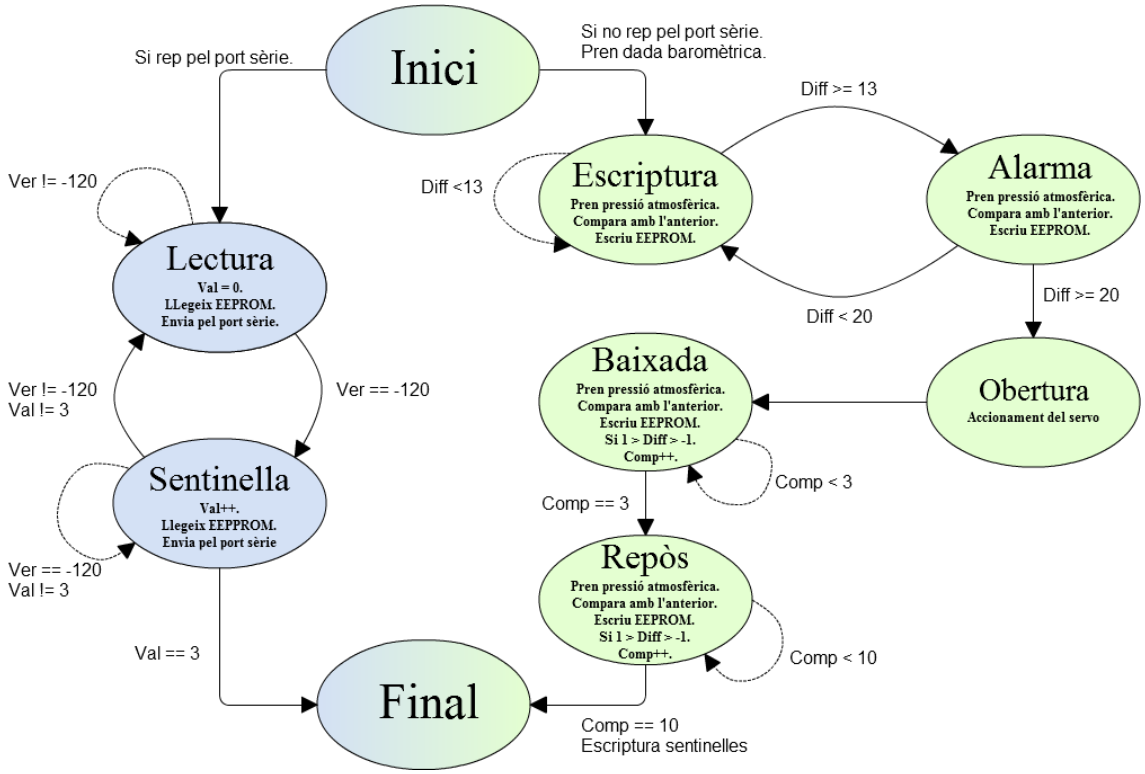
El següent estat anomenat Obertura, conte el valor que confirma que realment s'està descendint, i en aquesta gràfica es correspon al punt més alt, tot i que no sempre es així. Com ja he explicat anteriorment. en aquest estat es on es dona l'ordre per a que el servomotor canviï de posició i alliberi el paracaigudes.

El penúltim estat es l'anomena't Baixada. Conte els valors de la baixada física del dispositiu, comença un valor després de l'obertura i sacava just quan comença a rebre valors estables.

L'últim estat s'anomena Aturada, ja que es correspon als últims valors rebuts quan el dispositiu ja esta en repòs perquè ha acabat el descens. El primer valor d'aquest estat es un valor després de l'estat anterior i el últim valor arriba quan han passat una sèrie de valors molt petits, per assegurar que realment s'ha acabat el descens.

6.5. MAQUINA D'ESTATS

Amb la maquina d'estats podem entendre millor el comportament del sistema vist fins ara de tot el codi de L'Arduino. I també ens servirà com un resum de l'execució del codi.



El primer que veiem clarament, es la diferencia de color per a la part d'escriptura i la de lectura. La primera part té un color verd, mentre que la segona és de color blau.

Començant per la primera bombolla hi trobem l'estat inici, on ens podem decantar cap a l'apartat d'escriptura si no rebem res per el port sèrie abans dels 10 segons ja estipulats, o si pel contrari rebem quelcom anirem a parar a l'apartat de lectura.

6.5.1. Part d'escriptura

Suposem que no rebem res pel port sèrie, i anem a parar a l'apartat d'escriptura. El primer que farem serà prendre una mesura, i la desarem per a poder-la comparar com a dada anterior.

Seguidament entrarem en el primer apartat, el d'escriptura, on tornem a prendre la pressió. Després fem el corresponent càlcul per a saber si estem baixant o pujant i desem aquest valor a la memòria EEPROM. Si aquest valor que es la diferencia entre mostres es menor que 13 significa que no ens hem d'alarmar entrant un altre cop en la mateixa part del codi. Sinó, passarem al següent estat que es el d'Alarma.

Dintre de l'estat Alarma tornem a prendre la pressió atmosfèrica, si la diferencia d'aquesta amb l'anterior es més gran de 20, la desem i entrem en el següent estat, el que significarà que

realment estem en caiguda lliure. Si pel contrari el valor de la diferencia es menor de 20 tornarem a l'estat anterior on seguirem llegint les dades pel rebudes pel port sèrie.

Un cop dintre de l'estat Obertura hem d'accionar el servo motor per a que alliberi ràpidament el paracaigudes. En aquest estat també es desa la pressió d'obertura per a després amb la pressió inicial i aquesta calcular l'altura d'obertura del paracaigudes.

Ara ja estem en l'estat baixada, això significa que el conjunt esta en descens amb el paracaigudes desplegat o desplegant-se. Anirem fent el mateix que en l'estat d'escriptura, anar comparant dades, però quan aquesta diferencia es situí entre 1 i -1 sumarem una posició a **comp**. Mentre s'estigui baixant, la diferencia de dades seran més grans de 1, i un cop que s'arribi al terra aquestes dades estaran al voltant de 1 i -1. Es per això que quan es rebin 3 dades entre aquests valor significarà que esta en repòs i per això es canvia d'estat.

L'últim estat dins d'escriptura es el de Repòs, que enregistrarà les dades fins que acabi rebent 7 valors petits mes. Quan la suma total sigui igual a 10, aleshores escriurà els sentinelles al final de la última posició de la memòria EEPROM i s'aturarà el programa.

6.5.2. Part de lectura

Si rebem alguna cosa pel port sèrie, entrarem a l'apartat de lectura, on aquesta part del codi disposa nomes de dos estats.

El primer dels estats, va llegint de la memòria EEPROM. Comença envia el valor llegit d'aquesta cap al port sèrie, i després comprova si el valor llegit es tracta d'un sentinella o no. En el cas de que el valor no sigui un sentinella, es a dir que el valor no sigui -120, tornarà a entrar dintre de l'apartat Lectura i continuarà llegint fins que el trobi.

Un cop trobat el sentinella passarà a l'estat Sentinella, on comprovarà si els següents valors també ho son. Si el següent valor després d'haver trobat un sentinella no ho és, tornarà al estat Lectura. Si pel contrari si que ho és tornarà a entrar en el mateix estat, fins que tingui 3 sentinelles seguits, és a les hores quan es donarà per finalitzat el programa.

6.6. PYTHON

Apart del propi codi creat per a l'Arduino, també he creat un petit codi en llenguatge de programació Python per a facilitar la lectura de dades. Aquest codi ha estat creat des d'un editor de codi anomenat Sublime Text.

Com ja he explicat, per tal de llegir les dades contingudes dintre de la memòria EEPROM, necessitàvem connectar l'Arduino al PC utilitzant un monitor sèrie que permetés veure els missatges que es reben per aquest port i també ser capaços d'enviar dades nosaltres per tal d'entrar en el programa de lectura.

Aquest programa en Python creat, ajuda en la lectura de dades ja que al executar-lo escolta el que es rep pel port sèrie, on es troba l'Arduino. Envia una dada per a entrar en el programa de lectura i finalment per a facilitar les coses desa la informació rebuda en un fitxer extern.

El codi del programa es el següent:

```
#!C:\Python27

# Importem la llibreria de PySerial
import serial

# Obrim el port corresponent al arduino i fixem el Baudrate a 9600
PortSerie = serial.Serial('COM9', 9600,)
# Creem fixer
text=open('Barometre.csv','w')
Arduino = PortSerie.readline()
print Arduino
PortSerie.write('l')


# Creem un bucle sense fi per tal d'emmagatzemar totes les dades rebudes
while True:
  # llegim fins a trobar el final de línia
  Arduino = PortSerie.readline()
  # obrim el fitxer
  text=open('Barometre.csv','a')
  #desem el contingut llegit
  text.write(Arduino[:-1])
  #mostrem per pantalla
  print Arduino
  #tanquem el fitxer
  text.close()
```

Primerament necessitem importar la llibreria serial per a poder-nos comunicar a traves d'aquest port. Seguidament obrirem el port corresponent al nostre Arduino, en aquest cas el COM9 des de Windows, amb un baud rate de 9600.

El següent pas consisteix en crear el fitxer en el qual volem desar les nostres dades, en aquest cas es tracta de un arxiu amb extensió .csv, que es un fitxer d'Excel per a facilitar la creació de gràfiques i el tractament de les dades. A l'hora d'obrir el fitxer es fa en mode "w", ja que en aquest mode si el fitxer no existeix el crea, i si per el contrari ja existeix, sobre escriu al damunt esborrant les dades ja existents.

A continuació llegim el que ens arribi pel port sèrie, per a establir-hi comunicació però sense desar-ho enlloc, nomes o mostrem per pantalla. Això es fa per tal de que el canal de comunicació s'obri. El proper a fer és enviar pel port sèrie un caràcter per tal que l'Arduino reconegui que es vol entrar en el programa de lectura.

Després d'entrar en el programa de lectura, toca anar llegint les dades rebudes pel port sèrie i anar emmagatzemant cada una dins del fitxer creat anteriorment. Aquest procés es fa de la següent manera:

- Llegim del port sèrie, i desem el contingut en una variable anomenada Arduino.
- Obrim el fitxer en mode "a", per tal d'escriure al final del fitxer existent.
- Escrivim el text rebut anteriorment pel port sèrie, i el desem dintre del fitxer.
- Ho mostrem per pantalla per saber que és el que va guardant en cada moment.
- I finalment tanquem el fitxer.

Aquest procés és va repetint sempre que anem rebent dades pel port sèrie. Un cop ja no es rebin dades pel port sèrie, s'aturarà el programa deixant un fitxer amb el contingut de les dades de la memòria EEPROM.

7. CONSTRUCCIÓ I DISSENY DEL CIRCUIT IMPRÈS DEL PARACAIGUDES

Per a les proves inicials del dispositiu d'obertura, resulta realment còmode treballar amb una placa de prototipatge o protoboard. Es una manera rapida i senzilla de connectar els diferents elements que formen el conjunt: Arduino, sensor baromètric, memòria EEPROM, bateria... Aquetes connexions és fan mitjançant l'ús de cables mascles.

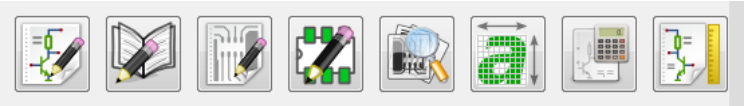
Però com tot també té els seus inconvenients. Resulta pràctic a l'hora de fer un disseny ràpid de la diferent distribució dels elements del circuit, com aniran connectats i veuren el funcionament, però es molt voluminós i impossible d'adossar al conjunt del paracaigudes, és en aquest punt quan decideixo crear una placa de circuit imprès per al paracaigudes.

Per a fer el disseny del circuit imprès, existeixen varis programes lliures i gratuïts. De tots els diferents programes disponibles, el que he utilitzat ha estat un anomenat KiCAD. Aquest es un programa que no havia fet servir mai abans, però amb l'ajuda de la documentació que hi ha per Internet sobre la seva utilització, es relativament fàcil aprendre sobre el seu funcionament.

El programa es pot trobar totalment gratuït i per a les diferents plataformes existents a la següent web: www.kicad-pcb.org. La plataforma que he utilitzat per a treballar ha estat Windows 7.

Un cop instal·lat el programa, el primer que s'ha de fer es crear un projecte nou, on es desaran tots els fitxers relacionats amb el projecte en qüestió. Fet aquest pas ja podem començar a dissenyar.

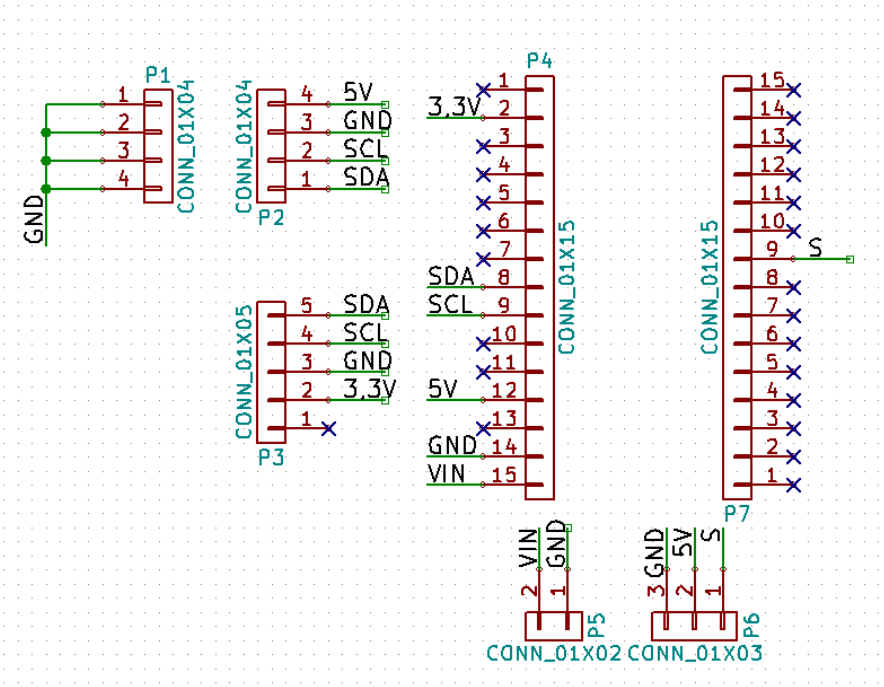
Cal dir que es un programa molt complet tot i ser de software lliure, per tant hi ha molts aspectes que desconec sobre el programa, però explicaré els més importants i els que he fet servir.



7.1. DIBUIX ESQUEMÀTIC

La primera icona, es correspon al disseny del circuit. Es a dir quins components en formaran part i com estaran connectats entre si.

Els components es poden triar des d'una llibreria de components que inclou el programa, on hi ha una llarga llista d'aquests. Passant des de simples connectors fins als microcontroladors.



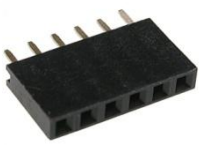
L'esquema anterior mostra els diferents elements que he utilitzat per a crear el circuit:

- P1 i P2 es corresponen a la memòria EEPROM.
- P3 és el sensor baromètric.
- P4 i P7 són les diferents entrades i sortides del Arduino Nano.
- P6 són les sortides del servo motor.
- P5 són els connectors per a la bateria.

Com es pot veure els diferents elements estan connectats mitjançant etiquetes en el cablejat del circuit. Si es posa el mateix nom a dos o més cables del circuit, aquests queden automàticament connectats entre si. També es poden connectar unint els cables directament entre si, però d'aquesta manera queda tot més net i endreçat.

Les X marcades serveixen per indicar que en aquella pota no hi ha res connectat.

En un principi volia utilitzar les diferents llibreries per a cada un dels elements del circuit, però després vaig optar per utilitzar nomes un tipus de llibreria que es el corresponent al que s'anomena en angles (socket strips).



D'aquesta manera si qualsevol dels components es vol utilitzar en un altre projecte es possible.

Un cop es tenen tot els elements connectat, s'ha de posar el nom a cada un d'ells. Quan els components es posen sobre l'esquemàtic reben una lletra majúscula que fa referència al component i un interrogant que més endavant es converteix en el nombre depenent dels elements que esdevinguin en l'esquemàtic.

Per exemple si tenim dos resistències apareixerien així: R? i R?, ja que la R és per a resistències. si en canvi tenim un condensador trobaríem la C. El següent pas és allistar tots els components del circuit a la opció [Anotate schematic components] o enumeració dels components del esquemàtic, quedant R1 i R2.

Com es pot apreciar els elements del dibuix, s'anoten amb P? i un cop pressionat el boto els enumera automàticament des del numero 1 fins als nombre d'elements existents, en aquest cas 7.

El següent pas, consisteix en mirar que no hi hagi problemes en el circuit o no hi hagi cap component sense connectar. Per a fer això es passa un test d'errors situat en un icona amb forma de marieta. El nom que rep aquest test és, (Perform electrical rules check) o traduït, mirar que compleixi les regles elèctriques.

Un cop passat el test, s'ha d'associar cada un dels components amb la petjada que deixarà a la placa. Per a fer això s'ha d'executar el CvPCB i triar de entre la llista, l'opció més escaient per al component seleccionat.

Un cop fet tot lo anterior ja poden generar un arxiu .NET que conte totes les dades per a la realització de la placa i que necessitarem per a la pròxima fase del procés.

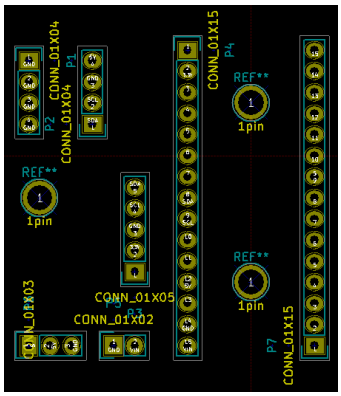
7.2. EDITOR DEL CIRCUIT IMPRÈS

Aquest és el programa on es fa el disseny de la distribució dels diferents components de la placa i de les pistes.

Dintre del menú principal, l'editor de circuits es correspon a la tercera icona.

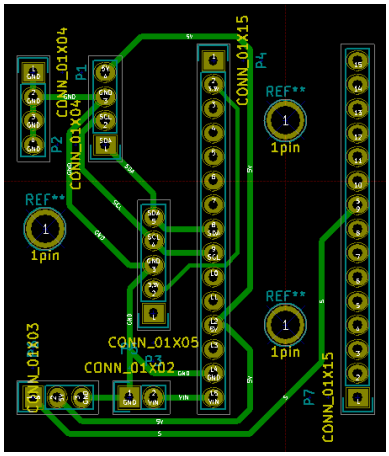
Un cop dintre de l'editor, el primer a fer es donar nom al document que tractarem, i seguidament importar l'arxiu .NET que s'ha creat en l'apartat anterior.

L'arxiu .NET conte la informació sobre tots els elements que hi haurà en el circuit. Ara es tracta de col·locar els elements de la manera que sigui més eficient o més optima per al propòsit que es necessiti. En aquest cas com el circuit va destinat a formar part del conjunt del paracaigudes ha de ser el més reduït possible.



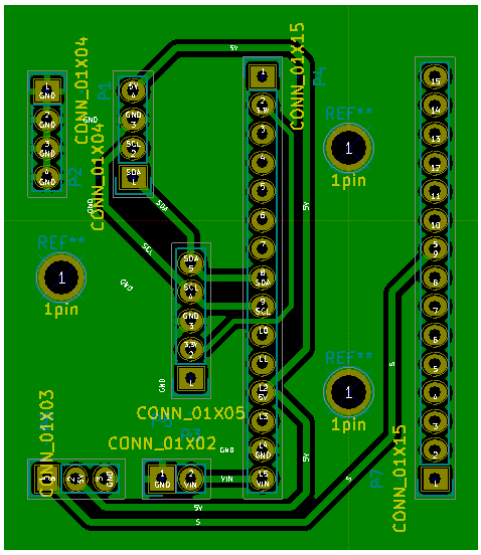
Com es pot apreciar els connectors estan posats a la distancia corresponent per a cada un dels elements. P1 i P2 formen la base per a la memòria EEPROM, mentre que P4 i P7 estan equidistàncies a les potes de l'Arduino Nano.

Un cop es tenen tots els elements on toquen, es hora de dibuixar les diferents pistes de coure per a connectar cada un dels pins amb el seu corresponent. Hi ha dos formes de fer-ho, o manualment dibuixant la pista a ma, o de forma automàtica. Utilitzant una eina del programa que relaciona els pins que estan associats, entre si mateixos. Es a dir tots els GND junts, 5V, etc.



Com podem apreciar a la imatge tots els pins necessaris ja estan connectats, però nomes ho estan per la capa de sota, també es podrien passar les línies de coure per sobre, però per a facilitar el procés de construcció de la placa, nomes passen per la capa de sota.

En el següent pas s'apliquen els marges de la placa, es a dir les dimensions que tindrà aquesta. Es tracta de dibuixar el contorn d'aquesta amb la capa anomenada "margin". Un cop dibuixat el contorn d'aquesta s'ha de triar un pla de massa.



El pla de masses es tria per connectar tots els pins que són el GND.

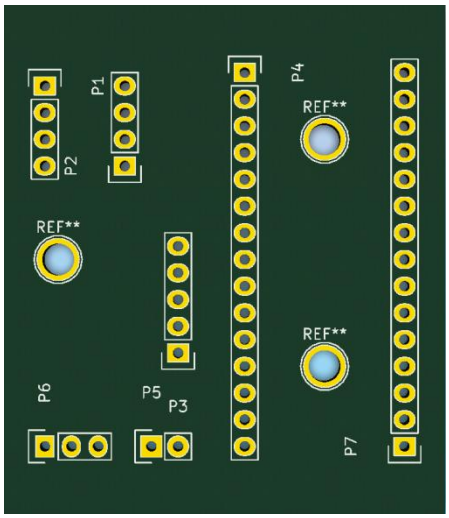
La part superior de la placa no conte coure, ja que no fa falta, mentre que la part inferior té una fina lamina d'aquest material. Al crear el pla de massa facilitem la feina a la fresadora, perquè nomes ha de passar per les parts que es veuen en negre, deixant al seu pas les diferents línies de connexió. D'altra manera la fresadora trigaria molt més a fer els circuits, perquè nomes deixaria les línies per a connectar els elements.

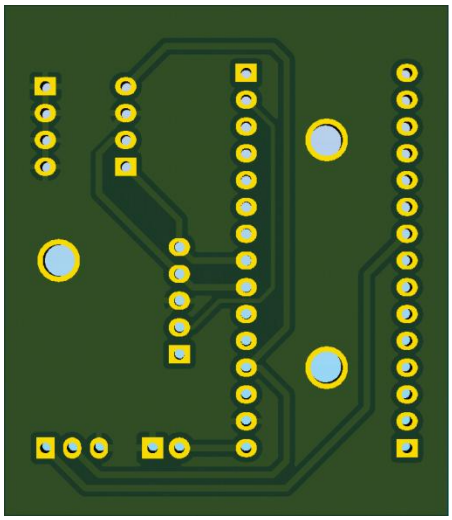
Un cop fet tot lo anterior, s'ha de verificar que tot sigui correcte passant el test de errors. Aquest mira que tots els pins estiguin connectats i si no hi ha res mal connectat, es pot procedir a obtenir els fitxers per a la fresadora.

Els fitxers que necessitem són tres:

- El primer dels fitxers es el de la grandària de la placa, es a dir el contorn. Aquest sobte des de l'apartat d'opcions i plot, seleccionant la capa 'margin' que es la del contorn. Si es prem el boto per guardar sens desarà a una ubicació per defecte. Aquest fitxer té l'extensió .gbr .
- El segon fitxer té a veure amb les pistes de coure. Es pot generar des del mateix apartat que el fitxer anterior, seleccionant la capa de coure pertanyent i així es genera el fitxer que indica a la maquina el gruix de les pistes de coure que s'han triat i per on han de passar.
- El tercer i últim fitxer que es necessita és el dels forats. Aquest fitxer ens indica la posició exacta d'on es troben els forats per a cada un dels components. Per accedir a realitzar aquest fitxer, s'ha d'anar al menú anterior i al costat del boto per a crear els fitxers anteriors, es troba el boto per a generar els forats. Aquest queda desat en el mateix directori que els anteriors fitxers, però en format .drl.

Ara que ja tenim els fitxers, podem anar a la nostra fresadora a fer la placa. I el resultat que obtindrem serà similar al següent:



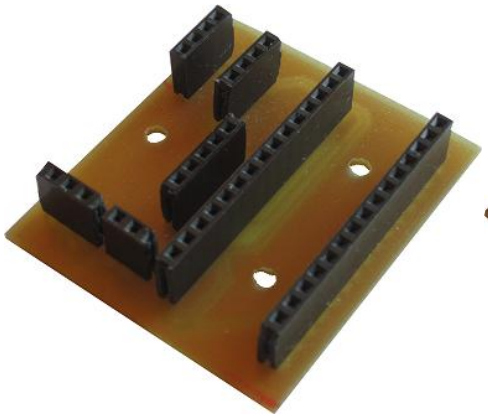


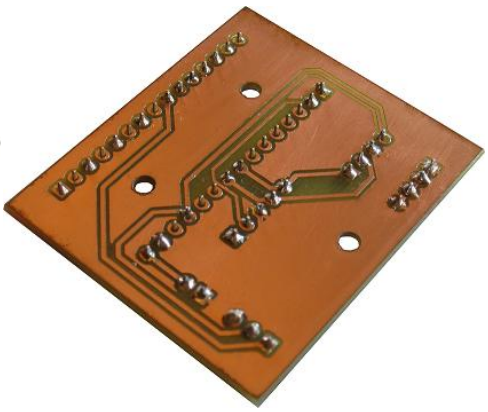
Podem observar que per la part de sobre nomes hi ha els forats corresponents als dispositius, mentre que per la part de sota, hi ha les diferents connexions entre els dispositius. Les parts més clares es correspon a les zones amb coure, mentre que les més fosques no contenen coure.

Per acabar aquest apartat vull afegir que el programa permet fer moltes més coses, com ara crear les llibreries per als teus components, crear la petjada que deixarà en el circuit imprès el component, entre d'altres. Però no han estat necessàries d'utilitzar per a la realització d'aquest projecte.

7.3. MUNTATGE DE LA PLACA

Un cop ja tinc la placa fresada, aquesta esta llesta per a que se li soldin els components. Com he comentat unes línies més amunt, no he soldat els components directament a la placa, sinó que per tal de poder reutilitzar-los en altres projectes, hi he soldat capçals per a poder inserir els components i retirar-los amb facilitat.





8. MUNTATGE DEL PROTOTIP

El muntatge del paracaigudes no ha estat una tasca fàcil, ja que molts dels elements utilitzats per a la realització del prototipus no estan dissenyats per complir el propòsit per al qual han estat utilitzats. Els elements que formen el conjunt del paracaigudes, la majoria san trobat en comerços locals o ja els tenia per casa, a excepció dels components electrònics que si que estan pensats per a complir la funció que fan.

Els elements que formen les diferents parts del paracaigudes s'han hagut d'adaptar per a que fessin la funció corresponent que els i tocava. Un exemple molt clar es la tela del paracaigudes que es tracta d'un vell paraigües de pesca força gran, tal i com s'explica en el apartat de la tela del paracaigudes.

Però d'alguna manera me les he arreglat per a construir el model que tenia en ment amb els diferents elements que tenia al meu abast.

La construcció del prototipus s'ha diferenciat bàsicament en dos parts:

- El paracaigudes, que és la part més mecànica del projecte, on la tela del paracaigudes o la molla en formen part.

- I la capsa de L'Arduino que es tracta de la part electrònica, on l'Arduino o el sensor baromètric en formen part entre d'altres.

Seguidament veurem en detall quin són cada un dels components que formen part de cada un dels conjunts.

8.1. PARACAIGUDES

Aquest és el conjunt més mecànic dels dos. S'explica les diferents parts que formen aquest conjunt i com ha estat realitzat el muntatge de tot plegat.

El primer que tenim és la tela del paracaigudes, la qual formava part d'un vell paraigües de pesca. Aquesta tela però, ha estat utilitzada després de veure que per la seva forma i dimensions era adequades per al fi que jo la volia tal i com s'explica a l'apartat de la tela del paracaigudes. La finalitat d'aquesta peça consisteix en reduir la velocitat de caiguda de tot el conjunt, per a minimitzar els danys que es pugin ocasionar en el Drone o el giny volador al que vagi adossat.

Un cop tenim la tela, es qüestió de provar-la i veure si realment funciona en el mon real. A cada un dels 8 vèrtex del paracaigudes se li ha lligat un cordill amb unes dimensions una mica superiors a la longitud del diàmetre del paracaigudes, per a que es desplegui completament i sigui efectiva. Al altre extrem de les cordes es va lligar un petit pes per a comprovar que realment frenaven la caiguda del objecte.

Cada una de les 8 cordes es capaç de suportar fins a 14kg de tensió, més que suficient per a qualsevol dels ginys al que es vulgui adossar.

El següent pas va consistir en buscar un recipient on poder desar el paracaigudes, i sigues suficientment gran per a que pogués sortir amb facilitat. El recipient es va trobar en un tot a 100 xines. Es tractava d'un recipient circular per a guardar la pasta de cuinar.

Amb el paracaigudes enrotllat, havia de comprovar si era fàcil per aquest sortir amb facilitat del recipient i no quedes aturat per a poder desplegar-se amb facilitat. Com que el pot es suficient ment lliscant em servia per al meu propòsit.

El problema que es presentava a continuació era trobar la manera per a extreure'l, amb prou força per a que aquest sortís completament del recipient. Així que la idea que sem va ocórrer va ser la de buscar una molla lo suficient ment gran per a fer aquesta funció.

El problema va venir a l'hora de trobar la molla ja que les ferreteries que vaig visitar no disposaven de les molles amb les característiques que estava buscant. Per la meva sort vaig trobar la molla que buscava en una caixa d'aquelles de broma, on el ninot de l'interior surt disparat al obrir la caixa. Aquesta caixa es va comprar en un comerç local i vaig aprofitar-ne la molla.

Seguidament tocava provar si la molla dintre del recipient tenia la força suficient per extreure el paracaigudes de dintre. La molla tenia tanta força que ni tan sols la tapa del propi recipient podia contenir-la. Però això no suposava un problema ja que la tapa s'avia que canviar per una de més resistent. La tapa que tenia en ment era un més dura, que pogués aguantar la força feta per a molla. Aquesta, la vaig extreure d'una llum d'escriptori espatllada, però va ser necessari tallar-la, llimar-la i polir-la per a que pogués formar part del conjunt.

Un cop avia trobat la tapa, i ja a les dimensions adequades, necessitava d'una frontissa per a poder obrir el compartiment i la tapa no marxes volant pels aires. La frontissa no va suposar un problema ja que és pot trobar en una ferreteria qualsevol. També vaig comprar cargols amb rosca de mètric 3 i 2 respectivament. Aquesta frontissa era de base plana pertant, la vaig haver de doblegar una mica per les puntes en una de les bases, per a que formes el contorn del cilindre on havia d'anar cargolada. L'altre base de la frontissa no era problema ja que era plana de la mateixa manera que la superfície de la tapa.

Ara ja tenia la molla, el paracaigudes, el recipient i la tapa. Però encara faltava lligar el paracaigudes al recipient i obrir la tapa quan cregués oportú. Per a lligar el paracaigudes al recipient vaig fer el següent:

- Primerament fer un forat a la base del paracaigudes on hi va un cargol, un tros de regleta, i la rosca.
- Fer passar la regleta pel cargol i lligar tres cordills a la regleta, de tal manera que els cordills quedin lligats a la base mitjançant la regleta. La longitud dels cordills, que són els mateixos que els utilitzats en la tela del paracaigudes, no cal que siguin massa llargs però si una mica més que la longitud de la molla.
- Seguidament tocava col·locar la molla dintre del cilindre, fent passar els cordills per l'interior de la molla. Al ser aquesta força ample necessitava una plataforma suficient ment resistent i lleugera, per evitar que el paracaigudes entres dintre de la pròpia molla i no es desplegues correctament degut a això. Aquesta forma havia

de ser similar a un cd amb les dimensions més reduïdes i més resistent. Com que disposava d'un vell disc dur vaig aprofitar el propi disc per a aquest propòsit, llimant-lo i polint-lo per fer les seves dimensions més reduïdes.

- El següent pas és unir els tres cordills lligats a la base del paracaigudes amb els vuit del propi paracaigudes. Per aconseguir aquest propòsit i aprofitant la pròpia base del disc dur, vaig crear un anell d'alumini polit al qual i vaig fer els forats per a passar els 11 cordills i que fos la peça d'unió entre paracaigudes i la base d'aquest.
- Ja nomes falta adossar la peça que farà que el paracaigudes s'obri en el moment escaient, aquesta peça es el servomotor. El problema principal per a unir el servo al cilindre, és que necessita d'una superfície plana per a poder posar-lo mentre que el recipient del paracaigudes es corbat. La base del disc dur tenia una circumferència dibuixada a la base que era la part que ocupava el propi disc on es desaven les dades. Aleshores amb una serra i una llima, vaig adaptar aquesta curvatura per a que encaixes amb la forma del cilindre quedant la part plana per fora, i la part corbada per dintre. Amb la part plana que queda ja si pot cargolar el servo, i d'aquesta manera queda el muntatge del conjunt mecànic fet.

8.2. CAPSA DE L'ARDUINO

La capsa de l'Arduino es com la he anomenat, però realment em refereixo a la part on estan els components electrònics del conjunt.

Primerament tots els elements van ser col·locats en una protoboard, i per entendre del seu funcionament era necessari provar-los. Un cop tots els elements funcionaven correctament, era hora de portar-los a un espai més reduït, per tal de incorporar-los posteriorment al cilindre del paracaigudes.

Aquesta no va ser una tasca fàcil, perquè necessitava d'una capsa de dimensions acceptables on mi càpigues:

- La font d'alimentació o pila de 9v.
- L'Arduino UNO, el qual esta molt be i és fàcil d'utilitzar però les seves dimensions són bastant grans, i ocupa casi tot l'espai.
- El sensor baromètric que tot i ser petit, el cablejat necessari per a connectar-lo a l'Arduino era força voluminós.
- I finalment el cablejat necessari per al servo motor, que tot i no estar el motor dintre els cables eren força voluminosos també.

Vaig trobar una capseta en un basar xines on hi cabien tots els elements descrits anteriorment, però tenia els seus defectes:

- El primer, de tots és que els elements estaven distribuïts per dins de qualsevol manera i no estaven subjectes a res.
- El segon, que el cablejat era força complicat i caòtic i no s'entenia res, tot i funcionar perfectament.

- El tercer, és tractava del material de la capsa. El cartró utilitzat en aquella caixa era resistent però no lo suficient per a adossar-lo al recipient del paracaigudes.
- El quart i final resultava que la capsa no tancava be, es a dir es podia tancar però s'obria amb facilitat. La solució va ser precintar la junta, on s'uneixen la tapa i la resta, amb cinta aïllant. Però a amb això va sorgir un nou problema el qual ara, no es podia obrir la capsa.

Mes endavant vaig decidir d'incorporar la memòria EEPROM per millorar les prestacions del conjunt, i l'antic sistema del cablejat dins de la caixa ja no em servia degut a que les connexions i2c es tenen que connectar en el mateix socket de l'Arduino.

A partir d'aquí hi va haver dos canvis importants:

- El primer va ser substituir l'Arduino. l'UNO es l'element que més volum aportava al conjunt i s'avia de substituir per un altre Arduino que oferís les mateixes prestacions però que el seu volum fos més reduït. L'Arduino Nano va ser la solució, ja que tenia les mateixes connexions i les dimensions respecte l'Arduino Uno es veien reduïdes a la meitat.

- El segon canvi important va ser el de substituir tot el voluminós cablejat, per una reduïda plaqueta de circuit imprès. Aquesta va ser dissenyada organitzant els elements de tal manera que quedessin endreçats i les dimensions fossin acceptables per al futur emplaçament de tot el conjunt el qual encara no estava decidit.

Cal dir que també hi habia la idea de substituir la bateria de 9v per dues bateries d'ió liti connectades en sèrie que aportaven cada una 3,7 volts i 2000mAh. Però no era viable ja que per a carregar-les sàvia de fer separadament, i aportaven un extra d'espai al emplaçament del conjunt electrònic amb la conseqüència de que no hi càpigues tot.

Per al moment la bateria de 9v és ideal ja que si s'acaba nomes s'ha de substituir per una altre, però això no es beneficiós per al medi ambient. Per aquest motiu lo ideal seria substituir la pila de 9v normal i corrent per una altre pila de les mateixes característiques i model però que fos recarregable. Allargant la seva vida útil i respectant el medi ambient.

Un cop solucionat el problema de les dimensions de l'Arduino i del cablejat, vaig començar a buscar una capsa per al emplaçament definitiu de les parts electròniques. Aquest emplaçament, vaig considerar que podria ser just a sota del cilindre en una caixa cilíndrica creada a partir d'una antiga llum de càmping, però no es va dur a terme.

Finalment vaig trobar una capsa per Internet que requeria les condicions necessàries:

- La part electrònica cabia perfectament, ja que estava feta expressament per a desar l'Arduino UNO, i les dimensions de la plaqueta amb tot muntat són inferiors.

- A més disposa d'un emplaçament especialment dissenyat per a desar la pila de 9v, perfecte per a alimentar tot el conjunt electrònic.

- Apart disposa de dos orificis pels quals es poden treure els cables que aniran al servomotor i també per a fer una medició més precisa del moment d'obertura.

Amb la caixa trobada nomes faltava col·locar-hi els elements.

La manera de fer que els elements quedin dintre de la capsa sense moure's va ser utilitzant lamines de goma EVA, que eviten que els components es moguin i també absorbeixen petites vibracions o impactes.

A la capsa en qüestió nomes he agut de fer una petita modificació, hi ha estat per a posar-hi l'interruptor que em permet reinicialitzar el sistema sense haver d'obrir la capsa cada vegada. Aquesta modificació s'ha realitzat fent un petit forat al costat per tal de posar-lo.

L'últim pas que restava, consistia en unir la part electrònica i la part mecànica. El problema consistia en que la superfície corbada del cilindre era difícil d'ajuntar amb la capsa del conjunt electrònic, ja que la base d'aquesta és plana. La solució que sem va ocórrer va ser la d'unir les dos parts utilitzant tires amples i llargues de velcro, tan amples i llargues com la capsa del dispositiu on esta la part electrònica.

és una solució que queda relativament forta, i hem permet desacoblar amb facilitat una part de l'altre. Així puc llegir les dades de la EEPROM amb facilitat.

El muntatge final queda de la següent manera:

9. L'EXPERIMENT

Per acabar el cos del text, es descriu un experiment realitzat amb tots els elements descrits fins ara es a dir el conjunt del paracaigudes.

9.1. DESCRIPCIÓ

Aquest experiment vol demostrar que el prototipus creat per tal de salvaguardar qualsevol giny volador, realment funciona i no nomes es tracta d'un concepte teòric.

L'experiment consisteix en llençar el paracaigudes des de un lloc suficientment alt, i comprovar si es capaç per si mateix d'adonar-se que esta caient i posteriorment desplegar el paracaigudes per evitar una caiguda aparatosa.

Després de descartar uns quants llocs per a dur a terme l'experiment, el lloc més elevat i amb millor zona d'aterratge va resultar ser aquest.

Es tracta d'un tram del pont de la variant, situat a Súria. L'altura d'aquest pont es de 28 metres des del terra fins a la part més alta.

A la part inferior del pont si pot trobar un petit camí asfaltat, on rarament i passen cotxes o vianants, mentre que al costat del camí hi ha vegetació força abundant, passant des d'arbres fins a males herbes. En altres paraules es una zona on el paracaigudes es pot desplegar amb la tranquil·litat de no molestar a ningú.

Pel que fa a la part superior del pont, es tracta d'un tram de la carretera C-55. Per això s'ha d'anar amb compte a l'hora de accedir-hi, i sempre amb l'armilla reflectant homologada per a que els cotxes et pugin veure en tot moment.

9.2. TEORIA

La teoria del experiment es la següent:

- Primerament es llença el paracaigudes des del pont, fent una petita paràbola cap endavant.
- Després al entrar en caiguda lliure hauria de detectar que es troba caient.
- Seguidament obrir el compartiment del paracaigudes, enviant una ordre de posició al servomotor.
- Finalment hauria de desplegar el paracaigudes i que tot el conjunt tingues un aterratge suau.

9.3. PRACTICA

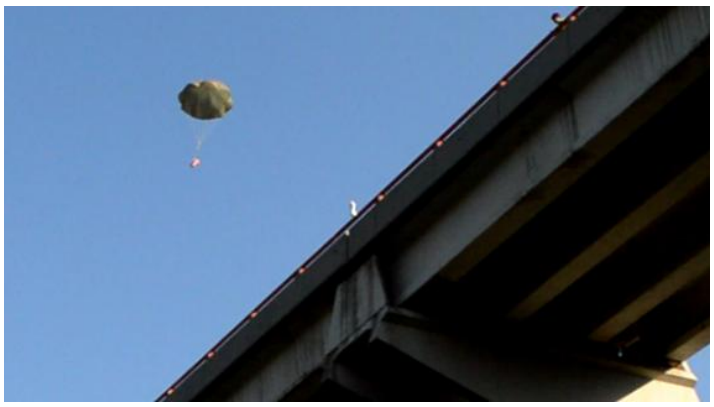
El experiment es farà en 2 parts:

- La primera part consisteix en llançar el paracaigudes sense el dispositiu d'obertura automàtic, amb la finalitat de veure el comportament que tindrà la part mecànica per si sola, i amb l'altura de 28 metres veure si es capaç de desplegar-se a temps. Si passa aquesta prova, es procedirà a la segona part.
- La segona part consisteix en fer el mateix experiment però ara amb el dispositiu d'obertura adossat al costat del paracaigudes, comprovant l'efectivitat del dispositiu d'obertura automàtic, veient així si realment funciona i que recull la memòria EEPROM.

9.4. PRIMERA PART

En aquesta primera part, tal i com he dit, el conjunt del paracaigudes nomes disposa de la part mecànica. El servomotor que normalment es el que retè la tapa del paracaigudes, no esta en la posició corresponent, i el que retè la tapa per a aquesta prova soc jo.

Tant bon punt deixi anar el paracaigudes aquest hauria de desplegar-se i inflar el paracaigudes.



I així va succeir, tant bon punt vaig llençar el paracaigudes cap avall del pont, aquest és va inflar ràpidament reduint dràsticament la velocitat de caiguda. Des del llançament fins a l'obertura d'aquest amb prou feines va transcórrer un segon.

La baixada va transcórrer amb normalitat i va durar al voltant d'uns 20 segons. Si tenim en compte que l'altura del pont es de 28 metres, podem calcular la velocitat mitjana a la que va descendir.

$$v_m = \frac{28\ metres}{20\ segons} = 1{,}4\ m/s$$

Aproximadament a 1,4 metres per segon. Si la part mecànica no funciones com pertoca, hauria caigut en caiguda lliure incrementant a cada segon la seva velocitat i resultant en una caiguda aparatosa i fatal.

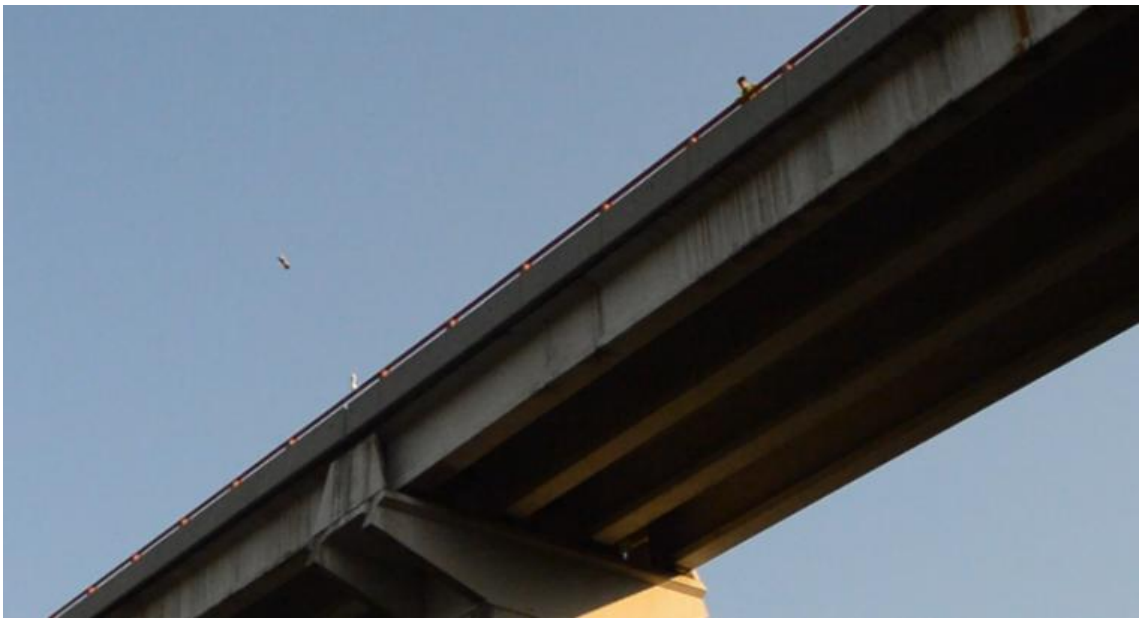
9.5. SEGONA PART

Després de comprovar que la part mecànica funciona com pertoca, es l'hora de fer la prova amb tot el conjunt sencer. Es a dir amb el dispositiu d'obertura automàtic adossat a un costat del cilindre del paracaigudes.

Aquesta prova es fa de la mateixa manera que l'anterior, però amb la diferencia que el conjunt ara s'ha d'obrir per ell mateix.

Al inici de la prova llenço el paracaigudes, formant una petita paràbola per tal de guanyar una mica més d'altura i temps. Quan el paracaigudes arriba al punt més alt al cap de 0,6 segons, comença el descens.

a partir d'aquest punt fins passat 1,24 segons, el paracaigudes no obre la tapa del paracaigudes alliberant aquest del seu interior.



En altres paraules, des del punt més alt fins a l'alliberament del paracaigudes ni tan sols transcorre un segon i mig. Es a dir que en aquest petit període de temps, ha tingut que rebre varies dades del sensor baromètric, adonar-se que estava en descens mitjançant els càlculs pertinents, donar la senyal al servomotor per a que anés a la posició corresponent i finalment aquest posar-se on pertoca ja que és un element mecànic que requereix de cert espai de temps.

El següent que succeeix és el desplegament complet del paracaigudes per tal de salvaguardar el giny volador.

En aquesta part del recorregut, es veu com el paracaigudes i el cilindre contenidor van en paral·lel això es degut a que falta pes en el contenidor. La grandària d'aquesta tela esta pensada per suportar un pes similar al del drone.

A mesura que baixa, el paracaigudes rep aire en el seu interior i permet el desplegament pertinent del paracaigudes.

Gairebé no s'aprecia degut a la llunyania del objecte però el cilindre del paracaigudes esta penjant de la tela del paracaigudes.

Gràcies a que el paracaigudes va complir amb la seva funció, el dispositiu va arribar a terra d'una sola peça amb un únic inconvenient, acabar caient a sobre d'uns esbarzers, però sense mes problema.

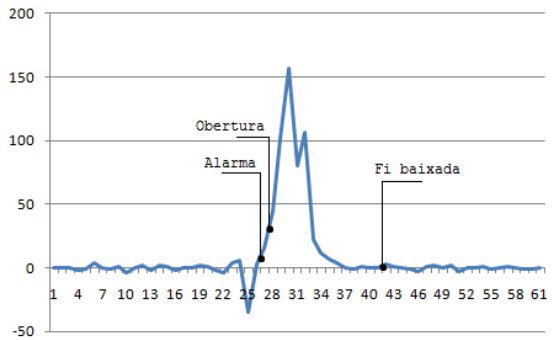
9.6. DADES OBTINGUDES

Ara anem a veure la part interessant del vol realitzat pel paracaigudes. Les dades que ha obtingut el sensor baromètric i posteriorment s'han desat en la memòria EEPROM.

Amb les dades obtingudes ens farem una idea de que ha passat durant el curt espai de temps que ha durat l'experiment.

Com ja sabem, les dades que es desen a la memòria EEPROM contenen la diferencia de pressió entre 2 mostres diferents, sabent així si l'aparell puja o baixa per a després determinar el moment d'obertura del paracaigudes.

Les dades obtingudes durant el vol formen la següent gràfica:



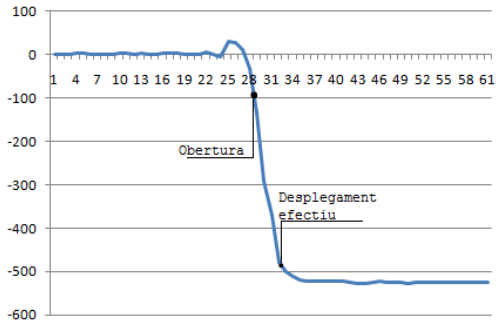
Quan la gràfica descendeix significa que la diferencia de pressió es negativa i per tant al ser el resultat negatiu la pressió disminueix i l'aparell esta pujant. Si la gràfica ascendeix, significa el

contrari que l'aparell esta descendint. Com més pronunciada es la gràfica significa que els canvis de pressió són més acusats i per tant la velocitat es major.

Els diferents estats pels que passa son:

- Inici: Va des de la primera dada fins a la anterior del estat Alarma.
- Alarma: Es la dada que indica que es descendeix a gran velocitat.
- Obertura: Quan s'arriba aquest estat definitivament s'està caient.
- Baixada: Compren totes les dades enregistrades durant la baixada.
- Repòs: Aquest estat s'inicia automàticament quan es considera que l'aparell esta aturat.

Si ara agafem els valors, els invertim i finalment els sumem entre ells, obtindrem una gràfica amb el perfil del vol del paracaigudes.



En aquesta gràfica podem observar el recorregut fet pel paracaigudes en funció del temps. On l'eix de les x es el temps transcorregut i el eix de les y és l'altura.

Observem que al principi esta a una mateixa altura, on es correspon a l'estat Inici, quan soc dalt del pont mentre espero el moment escaient fins que decideixo llençar-lo enlaire per a que agafi una mica més d'altura. Això es veu reflectit a partir de la dada 23, punt en el qual ja no el sostinc.

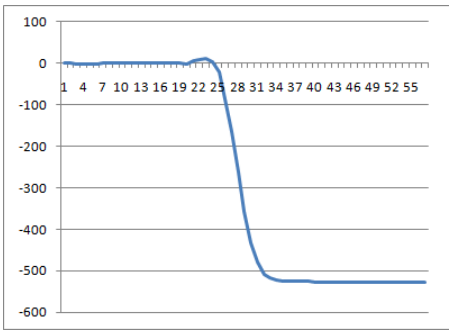
Després d'arribar al punt més alt, comença el descens del conjunt del paracaigudes. El punt més alt enregistrat es situa a la dada 25, mentre que l'obertura es situa a la dada 28. Sabent que entre dades han transcorregut 470ms, en altres paraules gaire be mig segon, l'obertura es realitza abans d'un segon i mig des de que comença el descens.

Seguidament podem apreciar com el conjunt descendeix lliurement fins al desplegament complet del paracaigudes que es produeix a la dada 32, que traduït a segons són menys de 2. A partir d'aquest punt s'observa com el conjunt redueix bruscament el seu descens i això es gracies a que el paracaigudes en redueix la velocitat.

Finalment, el conjunt tocarà terra i al cap de poc l'Arduino s'adonarà que ja no esta descendint gracies a les dades obtingudes pel sensor baromètric. En aquest punt es produirà una transició d'estat, passant de Baixada a Repòs.

Després d'obtenir suficients dades com per saber que el conjunt ja no esta en moviment, el programa s'atura automàticament posant fi a l'enregistrament de dades.

En el cas de que tinguéssim moltes dades a analitzar, podríem utilitzar una mitjana mòbil simple, que ajudaria a suavitzar les fluctuacions a curt termini, ressaltant així els cicles de llarg termini o tendència.



En aquest cas es tracta de la mitjana mòbil de 5 valors, mms5. Com es pot apreciar els valors tan del inici com del final són més constants i suavitzats.

10. CONCLUSIONS

L'objectiu d'aquest projecte, era desenvolupar un sistema que permetés salvar ginys voladors mitjançant el desplegament d'un paracaigudes. Es per això que he construït un prototipus per a demostrar la validesa del projecte.

No obstant, el prototipus és un element que s'ha fet artesanalment i en conseqüència no es perfecte. Un dels principals inconvenients d'aquest prototipus podria tractar-se del pes d'aquest, ja que té un pes aproximat de 650 grams, i no tots els ginys voladors són capaços d'enlairar-se amb aquest pes extra a sobre.

Un altre inconvenient a tenir en compte, és que el pes del prototipus no esta degudament centrat, ja que la part electrònica esta adossada al lateral del paracaigudes poden fer que l'aparell que l'enlairi es desestabilitzi.

Tot i així, aquest prototipus s'ha fet tenint en ment la protecció d'un Drone força gran. Capaç d'aixecar un pes força més elevat que aquest, i d'auto equilibrar-se per si mateix gracies al seu Microcontrolador. Per tant a priori això no suposaria un problema.

Un altre punt clau per a que funcioni adequadament, com ja hem vist en algun apartat, es necessari que el conjunt agafi certa altura, ja que la resposta mecànica del paracaigudes no es immediata.

Si el prototipus passes a ser un model comercial, seria indispensable millorar aquests petits inconvenients.

Primerament s'hauria de canviar els materials utilitzats per materials més resistents i de millor qualitat, encara que això fes que el preu del producte final augmentes significativament. Una opció seria canviar el plàstic i metalls utilitzats per fibra de carboni, i això ajudaria molt a disminuir el pes de tot el conjunt.

Gairebé tots els materials utilitzats es podrien beneficiar d'aquest canvi a exempció de la molla, que no podria ser de cap altre material que no fos l'actual. Però si que es podria canviar per una més eficient i de dimensions més reduïdes que complís el mateix propòsit.

La tela del paracaigudes podria ser canviada en funció de les necessitats del client. La grandària depèn del pes del giny volador que es vulgui salvaguardar tal i com ja hem vist, i també es podria canviar per una tela de millor qualitat tal com niló o seda augmentant però el preu del producte final.

Per tant es podria tenir de 2 a 3 models, començant pel més petit i per a drones de dimensions reduïdes, fins a un model més gran amb una gran tela com la construïda en aquest projecte per a grans ginys voladors.

El segon canvi a fer, hauria de ser centrar el pes del paracaigudes de tal manera que quedes el dispositiu d'obertura automàtica dintre del propi contenidor del paracaigudes. Això estaria be haver pogut dur-ho a terme, però per limitacions en la construcció no ha estat possible.

Tampoc resultava pràctic si volia accedir a la part electrònica del dispositiu. Es per això que aquesta part esta adossada al cilindre per a un fàcil accés.

En altres paraules la part mecànica del prototipus hauria de rebre aquests canvis si es volgués comercialitzar.

Pel que fa la part electrònica, si es volgués realitzar un model comercial la deixaria tal qual esta però aquest cop si amb els components soldats directament a la placa, i per abaratir costos en comptes d'utilitzar l'Arduino Nano, utilitzar nomes el propi Microcontrolador Atmega328.

En aquest projecte no s'ha realitzat perquè resulta molt més còmode treballar amb l'Arduino a l'hora de canviar els valors del codi i fer proves.

Una altre futura actualització, podria ser la d'implementar un mòdul bluetooth al conjunt, per poder enviar per exemple les dades a dispositius mòbils android, com si del port sèrie es tractes i poder conèixer les dades enviades pel port sèrie al moment.

Com a conclusió final del projecte, m'agradaria remarcar que ha estat un projecte en el qual m'ha agradat aprendre a treballar amb elements que no havia tocat mai tal com el sensor baromètric, la memòria EEPROM i la manera que té d'emmagatzemar les dades.

També m'ha agradat el fet de realitzar des de zero el meu propi circuit imprès: pensar-lo, dibuixar-lo amb l'ordinador i portar-lo al mon real.

Finalment m'agradaria acabar dient que si una cosa no surt de bones a primeres, amb una mica d'esforç i dedicació es pot aconseguir.

11. BIBLIOGRAFIA

11.1. A la xarxa:

Wikipedia (2015): Drag coefficient, recuperat el dia 15/4/2015, des de: http://en.wikipedia.org/wiki/Drag_coefficient

Wikipedia (2015): Aire, recuperat el dia 16/4/2015, des de: http://es.wikipedia.org/wiki/Aire

Wikipedia (2015): Beerware, recuperat el dia 5/4/2015, des de: http://en.wikipedia.org/wiki/Beerware

Angel Franco Garcia (2015): Descenso de un paracaidista, recuperat el dia 20/4/2015 des de: http://www.sc.ehu.es/sbweb/fisica/dinamica/paracaidista/paracaidista.html

F.R.R. Mallory (2015): Propiedades de las telas de poliéster, recuperat el dia 22/4/2015 des de: http://www.ehowenespanol.com/propiedades-telas-poliester-sobre_103789/

Arduino (2015): Arduino Nano, recuperat el dia 1/5/2015 des de: http://www.arduino.cc/en/Main/ArduinoBoardNano

Hobbytronics (2015): Adding External I2C EEPROM to Arduino, recuperat el dia 15/5/2015 des de: http://www.hobbytronics.co.uk/arduino-external-eeprom

Kicad (2015): Tutorial Kicad, recuperat el dia 20/5/2015, des de: http://ci.kicad-pcb.org/job/any-kicad-doc-head/lastSuccessfulBuild/artifact/src/Pcbnew/Pcbnew.html

Anonim (2015): Point to point, recuperat el dia 17/4/2015, des de https://docs.google.com/presentation/d/1Imwbhd_c2TdBoK5X5EU15tVkWJf8qcdAXyJ1yi3JlXA/embed?hl=en&size=s#slide=id.gc459e79_0_94

Bosch (2015): BMP180, recuperat el dia 1/4/2015, des de http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/Pressure/BMP180.pdf

Microchip (2015): 24AA32A/24LC32A, recuperat el dia 25/5/2015, des de http://ww1.microchip.com/downloads/en/DeviceDoc/21713M.pdf

Wikipedia (2015): Mitjana mobil, recuperat el dia 4/7/2015, des de https://ca.wikipedia.org/wiki/Mitjana_m%C3%B2bil

12. ANNEXOS

En els annexos hi ha descrit el codi del programa utilitzat, i el pressupost del que ha costat tot.

12.1. CODI DEL PROGRAMA DE L'ARDUINO

```
#include <SFE_BMP180.h> //llibreria utilitzada per al us del barometre
#include <Wire.h>       // llibreria utilitzada per al us de i2c
#include <Servo.h>      // llibreria utilitzada per al us del servomotor

#define disk1 0x50      //Addreça del xip eeprom 24LC32A

Servo myservo;        // creo objecte servo per controlar el servo

SFE_BMP180 pressure;

double presio, pres_ant, pres_res, mitj_pres=0, mitj_ant , baseline, altura, ober,
desa;

int pos = 0, x = 90, a = 0, obertura = 0, compt_fi = 0, lect_escr = 0;

uint8_t alt, alt2, altu;

int8_t ver = 0, val;

unsigned int address = 0, altura_rel, alt_baix, alt_fi, tura;

void setup() {
  Serial.begin(9600);                // inicialitzo el port serie
  Wire.begin(); //Utilitzat per iniciar el bus i2c

  Serial.println ("Introdueix qualsevol tecla per inicialitzar programa de lectura.");
  Serial.println ("Esperant tecla...");

  delay(5000);
  Serial.print ("5..");
  delay(1000);
  Serial.print ("4..");
  delay(1000);
  Serial.print ("3..");
  delay(1000);
  Serial.print ("2..");
  delay(1000);
  Serial.println ("1");
  delay(1000);

  if (Serial.available() > 0)
  {
  lect_escr = 1;
  Serial.println ("Entrant al programa de lectura.");
  delay(1000);
  }

  else
  {

  Serial.println ("Entrant al programa d'escriptura.");
  lect_escr = 2;
  myservo.attach(6);
  myservo.write(pos);

  if (pressure.begin())
    Serial.println("Barometre funcionant!");

  else
  {
    Serial.println("Quelcom a fallat, surt i torna a entrar.");
    while(1);
  }
  pres_ant = getPressure();
  presio = pres_ant;
  baseline = pres_ant;
  mitj_pres = getPressure();
```

```
  baseline = mitj_pres + baseline;
  mitj_pres = getPressure();
  baseline = (mitj_pres + baseline)/3;
  mitj_ant = baseline;
  }
}

void loop()
{

  if(lect_escr == 2)
  {
  digitalWrite(13, LOW);
  delay(58);

  presio = getPressure();     // 1a mostra
  mitj_pres = presio;
  pres_res = (presio - pres_ant) * 100;
  writeEEPROM(disk1, address, pres_res);
  address++;
  pres_ant = presio;
  delay(58);

  presio = getPressure();     // 2a mostra
  mitj_pres = presio + mitj_pres;
  pres_res = (presio - pres_ant) * 100;
  writeEEPROM(disk1, address, pres_res);
  address++;
  pres_ant = presio;
  delay(58);

  presio = getPressure();     // 3a mostra
  mitj_pres = presio + mitj_pres;
  pres_res = (presio - pres_ant) * 100;
  writeEEPROM(disk1, address, pres_res);
  address++;
  pres_ant = presio;
  delay(58);

  presio = getPressure();     // 4a mostra
  mitj_pres = presio + mitj_pres;
  pres_res = (presio - pres_ant) * 100;
  writeEEPROM(disk1, address, pres_res);
  address++;
  pres_ant = presio;
  delay(58);

  presio = getPressure();     // 5a mostra
  mitj_pres = (presio + mitj_pres)/5;
  pres_res = (presio - pres_ant) * 100;
  writeEEPROM(disk1, address, pres_res);
  pres_ant = presio;


  pres_res = (mitj_pres - mitj_ant) * 100;
  Serial.println(pres_res);
  desa = mitj_pres;
  digitalWrite(13, HIGH);

  if(pres_res>=13 && obertura == 0)
  {
    address++;
    presio = getPressure();
    pres_res = (presio - mitj_ant) * 100;
    writeEEPROM(disk1, address, pres_res);
    Serial.print("Pressio > 10: ");
    Serial.println(pres_res);
    if(pres_res>=20 && obertura == 0)
    {
      myservo.write(x);
      Serial.println("Paracaigudes obert");

      ober = pres_res;
      tura = address;

      altura = getPressure();
      mitj_pres = altura;
```

```
      altura = getPressure();
      mitj_pres = mitj_pres + altura;
      altura = getPressure();
      altura = (mitj_pres + altura)/3;

      obertura=1;
    }
  }
  if (obertura == 1)
  {
    if (1 > pres_res && pres_res > -1)
      {
        compt_fi++;

        if(compt_fi == 3)
        {
        alt_baix = address;
        }

        if(compt_fi == 10)
        {

        alt_fi = address;
        address++;
        writeEEPROM(disk1, address, -120);
        address++;
        writeEEPROM(disk1, address, -120);
        address++;
        writeEEPROM(disk1, address, -120);
        a = pressure.altitude(altura,baseline);

        Serial.print("Altura d'obertura respecte del terra: ");
        Serial.print(a,1);
        Serial.println(" metres.");

        address++;
        writeEEPROM(disk1, address, a); // altura en metres fins a 255
        a = a/256;
        address++;
        writeEEPROM(disk1, address, a); // voltes que san donat
        address++;
        writeEEPROM(disk1, address, ober);
        address++;
        writeEEPROM(disk1, address, tura);
        tura = tura/256;
        address++;
        writeEEPROM(disk1, address, tura);

        address++;
        writeEEPROM(disk1, address, alt_baix);
        alt_baix = alt_baix/256;
        address++;
        writeEEPROM(disk1, address, alt_baix);

        address++;
        writeEEPROM(disk1, address, alt_fi);
        alt_fi = alt_fi/256;
        address++;
        writeEEPROM(disk1, address, alt_fi);


        myservo.write(160);
        digitalWrite(13, LOW);
        while(1);
        }
      }

  }
  mitj_ant = desa;
  pres_ant = presio;
  address++;
  }


  else if(lect_escr == 1){
  val = 0;
  ver = readEEPROM(disk1, address);
```

```
  if (ver >= 0.00) Serial.print(" ");
  Serial.println(ver);
  if (ver == -120)
  {
    address++;
    val++;
    ver = readEEPROM(disk1, address);
    if (ver >= 0.00) Serial.print(" ");
    Serial.println(ver);
    if (ver == -120)
    {
      address++;
      val++;
      ver = readEEPROM(disk1, address);
      if (ver >= 0.00) Serial.print(" ");
      Serial.println(ver);
      if (ver == -120)
      {
        val++;
        if (val == 3)
        {
          address++;
          alt = readEEPROM(disk1, address);
          Serial.println(alt);
          address++;
          alt2 = readEEPROM(disk1, address);
          Serial.println(alt2);
          altura_rel = alt + (alt2 * 256);
          Serial.print("L'altura d'obertura respecte del terra es de ");
          Serial.print(altura_rel);
          Serial.println(" metres.");
          address++;

          altu = readEEPROM(disk1, address);
          address++;
          alt = readEEPROM(disk1, address);
          address++;
          alt2 = readEEPROM(disk1, address);
          altura_rel = 1 + alt + (alt2 * 256);

          Serial.println("Llegenda de la gràfica: ");
          Serial.println("Es descriu la posició dels valors");
          Serial.print("Inici ( 1 - ");
          Serial.print(altura_rel - 2);
          Serial.println(" )");
          Serial.print("Alarma ( ");
          Serial.print(altura_rel - 1);
          Serial.println(" )");

          Serial.print("Obertura ( ");
          Serial.print(altura_rel);
          Serial.print(" ) amb valor ");
          Serial.println(altu);

          Serial.print("Baixada ( ");
          Serial.print(altura_rel + 1);
          Serial.print(" - ");

          address++;
          alt = readEEPROM(disk1, address);
          address++;
          alt2 = readEEPROM(disk1, address);
          altura_rel = 1 + alt + (alt2 * 256);

          Serial.print(altura_rel);
          Serial.println(" )");

          Serial.print("Aturada ( ");
          Serial.print(altura_rel + 1);
          Serial.print(" - ");

          address++;
          alt = readEEPROM(disk1, address);
          address++;
          alt2 = readEEPROM(disk1, address);
          altura_rel = 1 + alt + (alt2 * 256);
```

```
          Serial.print(altura_rel);
          Serial.println(" )");
          while (1);
          }
      }
    }
  }
  address++;
  }

  else{
  Serial.println ("Hi ha agut algun error reinicia el programa");
  while(1);
  }

}

double getPressure()
{
  char status;
  double T,P,p0,a;

  // Primer s'ha de prendre la temperatura per a saber la pressió.
  // Comença la medició de la temperatura
  // Si te exit, el nombre de ms es retornat.
  // Sinó es retorna un 0 .

  status = pressure.startTemperature();
  if (status != 0)
  {
    // Esperem a que es completi la mesura:

    delay(status);

    // Recuperem el mesurament de la temperatura completada:
    // La Temperatura mesurada es guarda a T.
    // Utilitzo '&T' per a proveir la adreça de T a la funció.
    // La funció retorna 1 si te èxit, 0 sinó en te.

    status = pressure.getTemperature(T);
    if (status != 0)
    {
      // Mirem la pressió:
      // El paràmetre es l'ajust de mostreig, de 0 a 3 (highest res, longest wait).
      // Si la petició te exit, el nombre de ms a esperar es retornat.
      // sinó es retorna un 0.

      status = pressure.startPressure(3);
      if (status != 0)
      {
        // Esperem a que es completi la mesura:
        delay(status);

        // Recuperem el valor de la pressió efectuada
        // Guardem el valor a la variable P
        // Necessitem la temperatura T
        // Si la temperatura es estable, es pot fer una mesura de temperatura per una
        // sèrie de mesures de pressió.
        // Si es te exit es torna un 1 sinó un 0.

        status = pressure.getPressure(P,T);
        if (status != 0)
        {
          return(P);
        }
        else Serial.println("Error al calcular la temperatura reinicia el sistema\n");
      }
      else Serial.println("Error al calcular la temperatura reinicia el sistema\n");
    }
    else Serial.println("Error al calcular la temperatura reinicia el sistema\n");
  }
  else Serial.println("Error al calcular la temperatura reinicia el sistema\n");
}

void writeEEPROM(int deviceaddress, unsigned int eeaddress, int data ) //Escriure
{
  Wire.beginTransmission(deviceaddress);
```

```
  Wire.write((int)(eeaddress >> 8));   // MSB (Most Signifacant Bit)
  Wire.write((int)(eeaddress & 0xFF)); // LSB (Less Significant Bit)
  Wire.write(data);
  Wire.endTransmission();
  delay(10);
}

int8_t readEEPROM(int deviceaddress, unsigned int eeaddress )         //Llegir
{
  int8_t rdata;

  Wire.beginTransmission(deviceaddress);
  Wire.write((int)(eeaddress >> 8));   // MSB
  Wire.write((int)(eeaddress & 0xFF)); // LSB
  Wire.endTransmission();

  Wire.requestFrom(deviceaddress,1);

  if (Wire.available()) rdata = Wire.read();

  return rdata;
}
```

12.2. PRESSUPOST

En aquest apartat detallaré el preu de cada un dels elements utilitzats, i finalitzant per la part mecànica:

- Sensor Baromètric: 13€
- Pila: 1,5€
- Arduino NANO: 40€
- memòria EEPROM: 3,5€
- Servomotor : 10€
- Capsa Arduino: 10€
- Molla: 5€
- Cordills: 3€
- Cargols: 2€
- Tela del paracaigudes: 13€
- Recipient del paracaigudes: 1€
- Goma EVA: 1€

Tot els elements fan un total de 113€, però hi ha coses que no han estat comptades tal com piles de recanvi, memòria EEPROM extra o capses anteriors.

12.3. TEMPS INVERTIT

Calcular el temps invertit en el projecte no es una cosa fàcil, ja que no he comptat exhaustivament les hores dedicades, però hi ha hagut hores per a tot.

- Temps de pensar el projecte.
- Buscar informació a la xarxa.
- Buscar els materials.
- Entendre nou programari.
- Dissenyar la placa.
- Tallar, llimar, polir, muntar i soldar els components.

- Programar el codi.
- Fer les proves del prototip.
- Escriure la memòria
- Estudiar la viabilitat del projecte.